10ქ

საბოლოო მოწყობილობების, +

წვდომის კონტროლების, +

კიბერუსაფრთხოების როლების +

სოციალური ინჟინერიის +

3ქ

შიფრაციასთან +

CIA ტრიადა +

მავნე პროგრამული უზრუნველყოფები +

2ქ

**CIA ტრიადა - 3ქ**

**კონფიდენციალურობა** -ინფორმაციის საიდუმლოების ინდიკატორის, რომელიც უზრუნველყოფს ინფორმაციის დაცვას უკანონო გამჟღავნებისგან. ყველაზე ხშირად, უსაფრთხოების დარღვევა ხდება ინფორმაციის ავტორიზებული წვდომის მქონე პირის მიერ დაშვებული შეცდომის შედეგად.

**მთლიანობა** -ინფორმაციის, ინფორმაციული აქტივის სისწორისა და სისრულის მახასიათებელი;

**ხელმისაწვდომობა** - უფლებამოსილი პირის/ორგანიზაციის მოთხოვნის შესაბამისად აქტივზე წვდომის და გამოყენების მახასიათებელი.

აღნიშნული პრინციპები საერთაშორისო ასპარეზზე მოიხსენიება როგორც CIA Triad.

**მავნე პროგრამული უზრუნველყოფები - 3ქ**

მალვეარი არის მავნე პროგრამა რომელსაც დიდი ზიანის გამოწვევა შეუძლია.

არსებობს მავნე პროგრამების სხვადასხვა ტიპები:

ვირუსები: ვირსუები არის მავნე პროგრამების ერთ-ერთი ტიპი რომელიც შექმნილია

ისე რომ გავრცელდეს ერთი კომპიუტერიდან სხვა ბევრზე და ამას აკეთებს თავისი

თავის ლეგირიმურ ფაილებზე ან პროგრამებზე მიბმით.როგორც კი ვირუსი

დააინფიცირებს მოწყობილობას შეუძლია გამოიწვიოს დიდი ზიანი მაგალითად

წაშალოს ფაილები, მოიპაროს პირადი ინფორმაცია ან დაუბოკოს წვოდმა

მომხმარებელს დივაისზე და თვითონ აიღოს იგი.

ვირუსები შეიძლება სხვადასხვა გზით გავრცელდეს როგორიცაა usb, იმეილზე

მიმაგრებული ფაილები, საეჭვო ჩამოტვირთული ფაილები ინტერნეტიდან და სხვა.

ჭია: ჭია ანუ „ვორმი“ მალვეარის ერთ-ერთი ტიპია რომელიც ისე შექმნილი რომ

გავრცელდეს ერთი მოწყობილობიდან მეორეზე მეორედან მესამეზე და ა.შ

სისუსტეების გამოყენებით კომპიუტერულ სისტემაში ან ქსელში. ვირუსებისგან

განსხვავებით, რომელიც მოითხოვს რომ იუზერმა გახსნას ან გაუშვას ფაილი ,

ვორმები ავტომატურად ვრცელდებიან სისტემაში და ქსელში იუზერის ჩარევის

გარეშე. როგორც კი ვორმები სისტემაში მოხვდებიან მათ შეუძლიათ

გავრცელდნენიგივე ქსელში დაკავშირებულ სხვა სისტემებში და დიდი ზიანი

გამოიწვიონ, როგორიცაა ქსელის სისწრაფის და წარმადობის შემცირება,

სენსიტიური ინფორმაციის მოპარვა, ან მოწყობილობაზე კონტროლის მოპოვება.

ვორმები შეიძლება გავრცელდეს არასრული სოფთვეარის გამო, სუსტი პაროლებით

ან სოციალური ინჟინერიის შეტევებით, რომელიც ატყუებს და აიძულებს იუზერს

დააკლიკოს მავნე ლინკს ან ფაილს.

ტროას ცხენები(ტროიანები): ტროიანი მალვეარის ერთ-ერთი ტიპია რომელიც

შენიღბულია როგორც ლეგიტიმური პროგრამა ან ფაილი თუმცა გაშვებისთანავე

ახორციელებს მავნე ქმედებებს.

ტროიანებს შეუძლიათ მოიპარონ სენსიტიური ინფორმაცია, გათიშოს

უსაფრთხოების მახასიათებლები(ფიჩერები), ან დაუშვას არაავტორიზებული

წვდომა კომპიუტერზე ან ქსელზე. ისინი ასევე გამოიყენება რომ სხვა მალვეარის

გაავრცელოს კომპიუტერში როგორიცაა ვირუსები და ვორმები.

ტროიანები ძირითადად სოციალური ინჟინერიის შეტევებით ვრცელდება

როგორიცაა ფიშინგი ან ატყუებს მომხმარებელს რომ გადმოწეროს რაიმე ფაილის

ფეიკ ვებსაიტიდან. ტროიანები ასევე შეიძლება ისეთ ფაილებში ან სოფთვეარში

რომელიც ერთი შეხედვით ლეგიტიმური ჩანს.

ტროიანებისგან რომ თავი დავიცვათ საჭიროა მუდმივად განვაახლოთ ჩვენი

ანტივირუსი, მოვერიდოთ სოფთვეარის გადმოწერას უცნობი წყაროებიდან და

ყურადღებით ვიყოთ მოულოდნელ იმეილებთან და მესიჯებთან.

ლოგიკური ბომბი: ლოგიკური ბომბი არის მავნე კოდი ან პროგრამა რომელიც

შექმნილია ისე რომ გაეშვას მაშინ როცა რაიმე პირობა სრულდება. ეს პირობა

შეიძელბა იყოს გაკრვეული დრო ან საათი, ან რაიმე ივენთი, ან რაიმე მოქმედება

იუზერის მიერ. როდესაც ლოგიკური ბობმი გააქტიურდება მას შეუძლია წაშალოს

ფაილები ან მოიპაროს მნიშვნელოვანი დეითა, ან საერთოდ დაქრაშოს სისტემა.

სხვა მალვეარისგან განსხვავებით ლოგკური ბობმი ერთი სისტემიდან მეორეში არ

ვცელდება, პირიქით ისინი არიან სისტემაში მანამ სანამ რაიმე პირობა არ

შესრულდება.

ლოგიკური ბომბები ძირითადად დაინსტალირებულია ინსაიდერების მიერ,

როგორიცაა უკმაყოფილო თანამშრომელი რომელსაც სურს ზიანი მიაყენოს

კომპანიას.

„რენსომვეარი“: მალვეარის ერთ-ერთი ტიპია რომელიც გულისხმობს მსხვერპლის

ფაილების დაშიფვრას და მათი წვდომის შეზღუდვას სისტემასა და ფაილებზე,

მოკლედ რომ ვთქვათ დამნაშავეს მძევლად აჰყავს მსხვერპლის სისტემა და მასში

შემავალი ფაილების სანამ მსხვერპლი არ გადაიხდის გარკვეულ თანხას გარკვეულ

დროში.

გადახდა ძირითადად ხდება კრიპტოვალუტაში რომ ათაქერმა დამალოს თავისი

ვინაობა და ადგილმდებარეობა.

არსებობს ორი ტიპის რენსომვეარი, ესენია „ჩამკეტი რენსომვეარი“ და „კრიპტო

რენსომვეარი“. ჩამკეტი რენსომვეარი კეტავს იუზერს თავისი სისტემის გარეთ და

შეუძლებელს ხდის მისთვის სისტემაში შესვლას ან ფაილებზე წვდომას. კრიპტო

რენსომვეარი შიფრავს მსხვერპლის ფაილებს და ხდის მას ისეთს რომ ვერ წაიკიხავ

სანამ შიფრის კოდს არ მიიღებს მსხვერპლი ათაქერისგან.

გამოძალვითი მალვეარი ძირითადად ვრცდელდება ფიშინგის იმეილებით და მავნე

ჩამოტვირთვებით.როგორც კი მალვეარი მოხვდება სისტემაში, იგი იწყებს ფაილების

დაშიფვრას, კეტავს იუზერს სისტემის გარეთ და მის ეკრანზე აგდებს მალევეარის

ნოუთს რომ მისი ფაილების დაშიფრული და დალოქილია და მოუწევს თანხის

გადახდა.

ბექდორი(backdoor): ეს არის მალვეარი ტიპი რომელიც დამალულია სისტემაში ან

სოფთვეარში რომელიც ნებას რთავს ათაქერს მოიპოვოს არაავტორიზებული წვდომა

სისტემაზე ან სენსიტიურ ინფორმაციაზე. ამ მალვეარში განსაკუთრებული ისაა რომ

ათაქერის წვდომა სისტემაზე ისე ხორციელდება თითქოს წვდომა ავოტირზებულია

და მისი გამოვლენა ვერ ხერხდება.

ბექდორი შეიძება შეიქმნას განზრახ ან შემთხვევით. მაგალითად სოფთვეარ

დეველოპერმა შეიძლება შექმნას ბექდორი საკუთარი მიზნებისთბის რომ გატესტოს

ან დაადებაგოს აპლიკაცია მაგრამ თუ ათაქერმა იპოვა ეს ბექდორი მას შეუძლია

გამოიყენოს იგი საკუთარი მიზნებისთვის და მოიპოვოს არაავტორიზებული

წვდომა. ასევე ბექდორი შეიძლება შეიქმნას შემთხვევით პროგრამული ერორით

რაიმე სისუსტით აპლიკაციაში.

„რუთკიტი“: რუტკიტი მალვეარის ტიპია რომელიც შექმნილია ისე რომ დამალოს

თავისი არსებობა და მოქმედებები სისტემაში. რუტკიტები ძირითადად ათაქერების

ან მალვეარის მიერ ინსტალირდება და მიზნად ისახავს არაავტორიზებულ წვდომას

სისტემაზე და ამ წვდომის შენარჩუნებას იმის მიუხედავად ეს რუტკიდი იქნება

თუარა გამოვლენილი და წაშლილი სისტემიდან. ისინი ამას ოპერაციული სისტემის

მოდიფიცირებით ახდენენ რათა დამალონ თავიანთი არსებობა და მავნე აქტივობები

ანტივირუსებისგან და სხვა დაცვითი სოფთვეარებისგან.

რუტკიტები გამოიყენება სხვადასხვა მავნე საქმიანობისთვის როგორიცაა

სენსტიური ინფორმაციის მოპარვა, იუზერის მოქმედებების მონიტორინგი, სხვა

სისტემებზე თავდასხმა და ა.შ.

**შიფრაცია - 3ქ.**

**კრიპტოგრაფიული ალგორითმი ან შიფრა:** ეს ალგორითმი განსაზღვრავს დაშიფვრის და გაშიფვრის პროცესებს.

**გასაღები**: კრიპტოგრაფიულ ალგორითმს სჭირდება გასაღები, რათა გადაიყვანოს ჩვეულებრივი ტექსტი შიფრულ ტექსტად და პირიქით.

**plaintext** არის ორიგინალური შეტყობინება, რომლის დაშიფვრა გვინდა

**cybertext** ეს არის შეტყობინება მისი დაშიფრული ფორმით

**სიმეტრიული დაშიფვრის ალგორითმი** იყენებს იმავე გასაღებს დაშიფვრისა და გაშიფვრისთვის. შესაბამისად, კომუნიკაციის მონაწილე მხარეებმა უნდა შეთანხმდნენ საიდუმლო გასაღებზე, სანამ შეძლებენ რაიმე შეტყობინების გაცვლას. შემდეგ ფიგურაში, გამგზავნი უზრუნველყოფს დაშიფვრის პროცესს უბრალო ტექსტით და გასაღებით შიფრული ტექსტის მისაღებად. შიფრული ტექსტი ჩვეულებრივ იგზავნება რაიმე საკომუნიკაციო არხზე.

მეორეს მხრივ, მიმღები აწვდის გაშიფვრის პროცესს იმავე გასაღებით, რომელსაც გამოიყენებს გამგზავნი ორიგინალური ჩვეულებრივი ტექსტის აღსადგენად მიღებული შიფრული ტექსტიდან. გასაღების ცოდნის გარეშე, მიმღები ვერ შეძლებს ღია ტექსტის აღდგენას.

აქამდე ნახსენები ყველა ალგორითმი არის ბლოკშიფრული სიმეტრიული დაშიფვრის ალგორითმები. ბლოკის შიფრის ალგორითმი გარდაქმნის შენატანს (უბრალო ტექსტს) ბლოკებად და შიფრავს თითოეულ ბლოკს. ბლოკი ჩვეულებრივ 128 ბიტიანია. ქვემოთ მოცემულ ფიგურაში გვინდა დავშიფროთ ჩვეულებრივი ტექსტი „TANGO HOTEL MIKE“, სულ 16 სიმბოლოსგან. პირველი ნაბიჯი არის მისი ორობითი წარმოდგენა. თუ ვიყენებთ ASCII-ს, "T" არის 0x54 თექვსმეტობით ფორმატში, "A" არის 0x41 და ა.შ. ყოველი ორი თექვსმეტობითი ციფრი შეადგენს 8 ბიტს და წარმოადგენს ერთ ბაიტს. 128 ბიტიანი ბლოკი პრაქტიკულად 16 ბაიტია და წარმოდგენილია 4-ზე 4 მასივში. 128-ბიტიანი ბლოკი იკვებება როგორც ერთი ერთეული დაშიფვრის მეთოდით.

სიმეტრიული დაშიფვრის ალგორითმის სხვა ტიპია ნაკადის შიფრები, რომლებიც შიფრავს უბრალო ტექსტს ბაიტ-ბაიტზე. განვიხილოთ შემთხვევა, როდესაც გვინდა დავშიფროთ შეტყობინება „TANGO HOTEL MIKE“; თითოეული სიმბოლო უნდა გადაკეთდეს მის ორობით წარმომადგენლობაში. თუ ვიყენებთ ASCII-ს, "T" არის 0x54 თექვსმეტობით, ხოლო "A" არის 0x41 და ა.შ. დაშიფვრის მეთოდი ამუშავებს თითო ბაიტს. ეს წარმოდგენილია ქვემოთ მოცემულ ფიგურაში.

სიმეტრიული დაშიფვრა მომხმარებლებს მოითხოვს, რომ იპოვოთ უსაფრთხო არხი გასაღებების გაცვლისთვის. უსაფრთხო არხით, ჩვენ ძირითადად კონფიდენციალურობაზე და მთლიანობაზე ვზრუნავთ. სხვა სიტყვებით რომ ვთქვათ, ჩვენ გვჭირდება არხი, სადაც ვერც ერთი მესამე მხარე ვერ შეძლებს ტრაფიკის მოსმენას და წაკითხვას; მეტიც, გაგზავნილ შეტყობინებებსა და მონაცემებს ვერავინ შეცვლის.

ასიმეტრიული დაშიფვრა შესაძლებელს ხდის დაშიფრული შეტყობინებების გაცვლას უსაფრთხო არხის გარეშე; ჩვენ უბრალოდ გვჭირდება სანდო არხი. სანდო არხში ვგულისხმობთ იმას, რომ ჩვენ ძირითადად არხის მთლიანობაზე ვზრუნავთ და არა კონფიდენციალურობაზე.

**ასიმეტრიული დაშიფვრის** ალგორითმის გამოყენებისას, ჩვენ გამოვქმნით გასაღების წყვილს: საჯარო და კერძო გასაღებს. საჯარო გასაღები გაზიარებულია მსოფლიოსთან, უფრო კონკრეტულად, იმ ადამიანებთან, რომლებსაც სურთ ჩვენთან უსაფრთხოდ კომუნიკაცია. პირადი გასაღები უსაფრთხოდ უნდა იყოს შენახული და არავის არ უნდა მივცეთ მასზე წვდომა. უფრო მეტიც, საჯარო გასაღების ცოდნის მიუხედავად, კერძო გასაღების გამოყვანა შეუძლებელია.

როგორ მუშაობს ეს გასაღები წყვილი?

თუ შეტყობინება დაშიფრულია ერთი გასაღებით, მისი გაშიფვრა შესაძლებელია მეორე გასაღებით. Სხვა სიტყვებით:

თუ ალისა დაშიფვრავს შეტყობინებას ბობის საჯარო გასაღების გამოყენებით, მისი გაშიფვრა შესაძლებელია მხოლოდ ბობის პირადი გასაღების გამოყენებით.

პირიქით, თუ ბობი დაშიფვრავს შეტყობინებას მისი პირადი გასაღების გამოყენებით, მისი გაშიფვრა შესაძლებელია მხოლოდ ბობის საჯარო გასაღების გამოყენებით.

ჩვენ შეგვიძლია გამოვიყენოთ ასიმეტრიული დაშიფვრა კონფიდენციალურობის მისაღწევად შეტყობინებების დაშიფვრით მიმღების საჯარო გასაღების გამოყენებით. შემდეგ ორ ფიგურაში ჩვენ ვხედავთ, რომ:

ალისს სურს უზრუნველყოს კონფიდენციალურობა ბობთან კომუნიკაციაში. ის შიფრავს შეტყობინებას ბობის საჯარო გასაღების გამოყენებით, ხოლო ბობი შიფრავს მათ მისი პირადი გასაღების გამოყენებით. მოსალოდნელია, რომ ბობის საჯარო გასაღები გამოქვეყნდება საჯარო მონაცემთა ბაზაში ან მის ვებსაიტზე, მაგალითად.

ჩვენ გამოვიყენოთ ასიმეტრიული დაშიფვრა კონფიდენციალურობის შესახებ შეტყობინებების დაშიფვრით მიღწეული საჯარო გასაღების გამოყენება. შემდეგ ორ ფიგურაში ჩვენ ვხედავთ, რომ:

ალისს სურს უზრუნველყოს კონფიდენციალურობა ბობთან კომუნიკაციაში. ის შიფრავს შეტყობინებას ბობის საჯარო გასაღების გამოყენება, ხოლო ბობი შიფრავს მათ მისი პირადი გასაღების გამოყენება. მოსალოდნელია, რომ ბობის საჯარო გასაღები გამოქვეყნდება საჯარო მონაცემთა ბაზაში ან მის ვებსაიტზე, მაგალითად.

**RSA**

RSA მიიღო თავისი სახელი მისი გამომგონებლების, Rivest-ის, Shamir-ისა და Adleman-ისგან. იგი მუშაობს შემდეგნაირად:

აირჩიეთ ორი შემთხვევითი მარტივი რიცხვი, p და q. გამოთვალეთ N = p × q.

აირჩიეთ ორი მთელი რიცხვი e და d ისე, რომ e × d = 1 mod φ(N), სადაც ϕ(N) = N − p − q + 1. ეს ნაბიჯი საშუალებას მოგვცემს შევქმნათ საჯარო გასაღები (N) გასაღები (N,d).

გამომგზავნს შეუძლია დაშიფვროს მნიშვნელობა x გაანგარიშებით y = xe mod N. (მოდული)

მიმღებს შეუძლია გაშიფროს y x = yd mod N-ის გამოთვლით. გაითვალისწინეთ, რომ yd = xed = xkϕ(N) + 1 = (xϕ(N))k × x = . ეს ნაბიჯი განმარტავს, თუ რატომ ვაყენებთ შეზღუდვას e და d-ის არჩევანზე.

არ ინერვიულოთ, თუ ზემოთ მოყვანილი მათემატიკური განტოლებები ძალიან რთული ჩანდა; თქვენ არ გჭირდებათ მათემატიკა, რომ შეძლოთ RSA-ს გამოყენება, რადგან ის ადვილად ხელმისაწვდომია პროგრამებისა და პროგრამირების ბიბლიოთეკების საშუალებით.

RSA უსაფრთხოება ეყრდნობა ფაქტორიზაციას, რომელიც რთული პრობლემაა. მარტივია p q-ზე გამრავლება; თუმცა, შრომატევადია p და q მოცემული N-ის პოვნა. უფრო მეტიც, ეს რომ იყოს უსაფრთხო, p და q  საკმაოდ დიდი რიცხვები უნდა იყოს, მაგალითად, თითოეული 1024 ბიტიანი (ეს არის რიცხვი 300 ციფრზე მეტი). მნიშვნელოვანია აღინიშნოს, რომ RSA ეყრდნობა უსაფრთხო შემთხვევითი რიცხვების გენერირებას, ისევე როგორც სხვა ასიმეტრიული დაშიფვრის ალგორითმებს. თუ მოწინააღმდეგეს შეუძლია გამოიცნოს p და q, მთელი სისტემა ჩაითვლება დაუცველად.

განვიხილოთ შემდეგი პრაქტიკული მაგალითი.

ბობი ირჩევს ორ მარტივ რიცხვს: p = 157 და q = 199. ის ითვლის N = 31243.

ἕ(N) = N − p − q + 1 = 31243 − 157 − 199 + 1 = 30888, Bob selects= 30888, Bob selects=   3. აქ e × d = 163 × 379 = 61777 და 61777 mod 30888 = 1. საჯარო გასაღები არის (31243163) და პირადი გასაღები (31243379).

ვთქვათ, რომ დაშიფვრის მნიშვნელობა არის x = 13, შემდეგ ალისა გამოთვლის და გამოგიგზავნის y = xe mod N = 13163 mod 31243 = 16342.

ბობი გაშიფვრავს მიღებულ მნიშვნელობას x = yd mod N = 16341379 mod 31243 = 13 გამოთვლით.

**Hashireba**

კრიპტოგრაფიული ჰეშები და ჰეშირების ალგორითმები ფუნდამენტური ცნებებია კრიპტოგრაფიისა და ინფორმაციის უსაფრთხოებაში. კრიპტოგრაფიული ჰეშის ფუნქცია არის მათემატიკური ალგორითმი, რომელიც იღებს შეყვანის მონაცემებს (ან „შეტყობინებებს“) და აწარმოებს სიმბოლოების ფიქსირებული ზომის სტრიქონს, რომელიც, როგორც წესი, თექვსმეტობით რიცხვს წარმოადგენს. აქ მოცემულია კრიპტოგრაფიული ჰეშებისა და ჰეშირების ალგორითმების ძირითადი ასპექტები:

კრიპტოგრაფიული ჰეშები:

მიზანი:

მონაცემთა მთლიანობა: კრიპტოგრაფიული ჰეშები ჩვეულებრივ გამოიყენება მონაცემთა მთლიანობის უზრუნველსაყოფად. შეყვანის მონაცემების მცირე ცვლილებამაც კი უნდა გამოიწვიოს მნიშვნელოვნად განსხვავებული ჰეში.

ციფრული ხელმოწერები: ჰეშები გამოიყენება ციფრულ ხელმოწერებში დოკუმენტის ან შეტყობინების უნიკალური წარმოდგენის შესაქმნელად, რაც უზრუნველყოფს გამგზავნის ავთენტურობის გადამოწმების საშუალებას.

პაროლის შენახვა: ჰეშის ფუნქციები გამოიყენება პაროლების უსაფრთხოდ შესანახად. რეალური პაროლების შენახვის ნაცვლად, სისტემები ინახავს პაროლის ჰეშს და ამატებენ დაცვის ფენას.

კრიპტოგრაფიული ჰეშის ფუნქციების თვისებები:

განმსაზღვრელი: ერთი და იგივე შეყვანა ყოველთვის გამოიმუშავებს იგივე ჰეშის გამომავალს.

სწრაფი გამოთვლა: ჰეშის ფუნქცია უნდა იყოს ეფექტური გამოსათვლელად.

შეუქცევადობა: გამოთვლებით შეუძლებელი უნდა იყოს ჰეშის შებრუნება და ორიგინალური შეყვანის მიღება.

ფიქსირებული გამომავალი ზომა: ჰეშის გამომავალს აქვს ფიქსირებული ზომა, შეყვანის ზომის მიუხედავად.

ზვავის ეფექტი: შეყვანის მცირე ცვლილებამ უნდა გამოიწვიოს სრულიად განსხვავებული ჰეში

**Hashing Algorithms:**

1. **Common Cryptographic Hashing Algorithms:**
   * **MD5 (Message Digest Algorithm 5):** Although widely used in the past, MD5 is now considered weak due to vulnerabilities. It's not recommended for security-sensitive applications.
   * **SHA-1 (Secure Hash Algorithm 1):** Like MD5, SHA-1 has vulnerabilities, and its use in security-sensitive applications is discouraged.
   * **SHA-256, SHA-384, SHA-512 (Secure Hash Algorithm 2):** These SHA-2 variants are currently considered secure and are widely used for various security applications.
   * **SHA-3 (Secure Hash Algorithm 3):** The latest member of the Secure Hash Algorithm family, designed to provide an additional option for secure hashing.
   * **BLAKE2:** A cryptographic hash function that is faster than MD5, SHA-1, SHA-2, and SHA-3. It is designed to be fast and secure for a wide range of applications.
2. **Choosing a Hashing Algorithm:**
   * **Security Requirements:** The choice of hashing algorithm depends on the security requirements of the specific application. For most applications, SHA-256 or higher is recommended.
   * **Performance:** Consider the computational efficiency of the hashing algorithm, especially for applications with high throughput requirements.
   * **Compatibility:** Ensure compatibility with existing systems and standards.
3. **Cryptographic Hash Example (SHA-256):**

**MD5 (Message Digest Algorithm 5):**

* **Original Purpose:** MD5 was designed to produce a 128-bit hash value (32 hexadecimal characters) and was widely used for checksums and integrity verification.
* **Security Concerns:** MD5 is now considered cryptographically broken and unsuitable for further use due to vulnerabilities that allow collision attacks (different inputs producing the same hash).
* **Recommendation:** MD5 should not be used for security-sensitive applications, such as digital signatures or certificate generation.

**MD5 (Message Digest Algorithm 5):**

* **Original Purpose:** MD5 was designed to produce a 128-bit hash value (32 hexadecimal characters) and was widely used for checksums and integrity verification.
* **Security Concerns:** MD5 is now considered cryptographically broken and unsuitable for further use due to vulnerabilities that allow collision attacks (different inputs producing the same hash).
* **Recommendation:** MD5 should not be used for security-sensitive applications, such as digital signatures or certificate generation.

**. SHA-1 (Secure Hash Algorithm 1):**

* **Original Purpose: SHA-1, like MD5, was designed for integrity verification and produced a 160-bit hash value (40 hexadecimal characters).**
* **Security Concerns: SHA-1 is susceptible to collision attacks, where two different inputs produce the same hash. This vulnerability has led to its deprecation for security-sensitive purposes.**
* **Recommendation: SHA-1 should be avoided for cryptographic purposes, and SHA-2 or stronger algorithms should be used instead.**

**3. SHA-2 (Secure Hash Algorithm 2):**

* **Original Purpose: SHA-2 is a family of cryptographic hash functions, including SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, and SHA-512/256. SHA-256 and SHA-512 are the most commonly used variants.**
* **Security Features: SHA-2 is considered secure and is widely used for data integrity, digital signatures, and other cryptographic applications. It provides a higher level of security compared to MD5 and SHA-1.**

**Recommendation: For most security applications, especially those involving data integrity, SHA-256 or SHA-512 is recommended over MD5 and SHA-1**

**1. Use a Cryptographically Secure Hashing Algorithm:**

Choose a strong and secure hashing algorithm designed for password hashing. Common choices include:

* Argon2: Recommended by experts for password hashing due to its resistance to both GPU and ASIC attacks.
* bcrypt: A well-established algorithm with a configurable work factor, making it suitable for password hashing.
* scrypt: Designed to be memory-intensive, providing protection against certain types of attacks.

**2. Salt the Passwords:**

* Use a unique random salt for each password before hashing. A salt is a random value that is combined with the password before hashing, ensuring that even if two users have the same password, their hashed values will be different. This prevents attackers from using precomputed tables (rainbow tables) for password attacks**.**

**3. Use Key Stretching:**

* Key stretching involves applying the hash function multiple times (iterations) to slow down the hashing process. This adds an extra layer of security, making it more time-consuming and resource-intensive for attackers to perform brute-force or dictionary attacks.

**4. Implement Adaptive Hashing:**

* Choose a hashing algorithm that supports adaptive techniques, adjusting the complexity of the hashing process over time. This helps to mitigate the impact of advances in computing power and technologies.

**5. Regularly Update Hashing Algorithms:**

* Stay informed about the latest developments in cryptography and security. Periodically evaluate and update your password hashing algorithms to ensure they align with current best practices.

**6. Store Hashed Passwords Securely:**

* Store only the hashed passwords and associated salt values. Never store plaintext passwords or reversible encryption keys. Ensure that the storage mechanism for hashed passwords is itself secure to prevent unauthorized access.

**7. Keep Pace with Security Standards:**

* Stay updated on industry standards and recommendations for password hashing. Consider following guidelines from organizations such as NIST (National Institute of Standards and Technology) or OWASP (Open Web Application Security Project).

HMAC (Hash-based Message Authentication Code) is a mechanism used for authenticating the integrity and authenticity of a message or piece of data. It involves a cryptographic hash function and a secret key to produce a hash value that can be used to verify the integrity of the data. HMAC is commonly employed to ensure the integrity of databases and other sensitive information. Here's how you can use HMAC for authentication and ensure the integrity of databases:

**Using HMAC for Authentication:**

1. **Choose a Hash Function:**
   1. Select a secure cryptographic hash function, such as SHA-256 or SHA-3. The choice of hash function depends on your security requirements.
2. **Generate a Secret Key:**
   1. Generate a secret key that will be shared between the sender (database) and the receiver (application or user). This key should be kept confidential.
3. **Create the HMAC:**
   1. For each piece of data (e.g., a database record), calculate the HMAC using the selected hash function and the secret key. The formula for HMAC is:

**HMAC(key, message) = hash((key XOR opad) || hash((key XOR ipad) || message))**

where || denotes concatenation, opad is the outer padding, ipad is the inner padding, and hash is the chosen hash function.

1. **Attach the HMAC to the Data:**
   1. Append the calculated HMAC to the original data. This creates a combined message that includes the data and its HMAC.
2. **Send the Combined Message:**
   1. Send the combined message (data + HMAC) to the receiver.
3. **Verify at the Receiver's End:**
   1. At the receiver's end, recalculate the HMAC using the received data and the shared secret key. Compare this recalculated HMAC with the received HMAC. If they match, the data is considered authentic and hasn't been tampered with.

**Ensuring Integrity of Databases:**

1. **Apply HMAC to Database Records:**
   1. For each record in the database, apply the HMAC process as described above. Use a unique key for each record or a shared key for the entire database, depending on your security model.
2. **Store HMAC Values:**
   1. Store the calculated HMAC values along with the database records.
3. **Verification During Retrieval:**
   1. When retrieving records from the database, recalculate the HMAC for each record using the stored key. Compare the recalculated HMAC with the stored HMAC. If they match, the integrity of the record is intact.
4. **Protecting Against Tampering:**
   1. If an attacker attempts to tamper with the database by modifying records, the HMAC verification will fail, indicating that the data has been compromised.

By using HMAC, you can provide a strong guarantee of data integrity, ensuring that the database records have not been altered or tampered with. This is particularly important for applications that require secure and trustworthy storage of sensitive information.

**Using a key exchange such as the Diffie-Hellman key exchange** allows us to agree on a secret key under the eyes and ears of eavesdroppers. This key can be used with a symmetric encryption algorithm to ensure confidential communication. However, the key exchange we described earlier is not immune to Man-in-the-Middle (MITM) attack. The reason is that Alice has no way of ensuring that she is communicating with Bob, and Bob has no way of ensuring that he is communicating with Alice when exchanging the secret key.

Consider the figure below. It is an attack against the key exchange explained in the Diffie-Hellman Key Exchange task. The steps are as follows:

1. Alice and Bob agree on *q* and *g*. Anyone listening on the communication channel can read these two values, including the attacker, Mallory.
2. As she would normally do, Alice chooses a random variable *a*, calculates *A* ( *A* = (*ga*) mod *q*) and sends *A* to Bob. Mallory has been waiting for this step, and she has selected a random variable *m* and calculated the respective *M*. As soon as Mallory receives *A*, she sends *M* to Bob, pretending she is Alice.
3. Bob receives *M* thinking that Alice sent it. Bob has already picked a random variable *b* and calculated the respective *B*; he sends *B* to Alice. Similarly, Mallory intercepts the message, reads *B* and sends *M* to Alice instead.
4. Alice receives *M* and calculates *key* = *Ma* mod *q*.
5. Bob receives *M* and calculates *key* = *Mb* mod *q*.

Alice and Bob continue to communicate, thinking that they are communicating directly, unaware that they are communicating with Mallory, who can read and modify the messages before sending them to the intended recipient.

**სოციალური ინჟინერია - 10ქ.**

სოციალური ინჟინერია არის ტერმინი, რომელიც გამოიყენება ნებისმიერი კიბერშეტევის აღსაწერად, სადაც სამიზნე ადამიანია (და არა კომპიუტერი); ამ მიზეზით, მას ხანდახან მოიხსენიებენ, როგორც "ხალხის ჰაკერს". მაგალითად, თუ თავდამსხმელს სურს მიიღოს მსხვერპლის პაროლი, მას შეუძლია სცადოს პაროლის გამოცნობა ან უხეში ძალისხმევით — ან შეიძლება უბრალოდ გკითხოთ.

მიუხედავად იმისა, რომ ზემოთ მოყვანილი მაგალითი შედარებით მარტივია, სოციალური ინჟინერიის თავდასხმები შეიძლება გახდეს ძალიან რთული და ხშირად მოჰყვეს თავდამსხმელის მნიშვნელოვან კონტროლს სამიზნის ცხოვრებაზე - როგორც ონლაინ, ასევე ოფლაინზე. სოციალური ინჟინერიის შეტევები ხშირად მრავალფენიანია და ესკალაცია ხდება თოვლის ბურთის ეფექტის გამო. მაგალითად, თავდამსხმელმა შეიძლება დაიწყოს მცირე რაოდენობის საჯაროდ ხელმისაწვდომი ინფორმაციის მოპოვებით მსხვერპლის სოციალურ მედიაში ყოფნიდან, რომელიც შემდეგ მათ შეუძლიათ გამოიყენონ დამატებითი ინფორმაციის მისაღებად, მაგალითად, თქვენი ტელეფონის ან ფართოზოლოვანი პროვაიდერისგან. მეორე ეტაპიდან მიღებული ინფორმაცია შეიძლება გამოყენებულ იქნას უფრო სასარგებლო ინფორმაციის მოსაპოვებლად, შემდეგ კი ეტაპობრივად გადაიზარდოს მსხვერპლის საბანკო ანგარიშზე.

სოციალური ინჟინერიის სხვა ფორმები

ქარიზმატული ჰაკერები, რომლებიც ურეკავენ თქვენს სატელეფონო კომპანიას და ფლობენ თქვენს ანგარიშს, სოციალური ინჟინერიის ერთ-ერთი ფორმაა; თუმცა, არსებობს მრავალი სხვა ტიპი. სოციალური ინჟინერია არის ვრცელი თემა, რომელიც მოიცავს ნებისმიერ თავდასხმას, რომელიც ეყრდნობა ადამიანების მოტყუებას თავდამსხმელისთვის წვდომის მისაცემად და არა უშუალოდ ტექნოლოგიაზე თავდასხმაზე. მიუხედავად იმისა, რომ სამიზნეებთან პირდაპირი ურთიერთქმედება სოციალური ინჟინერიის ყველაზე გავრცელებული სტილია, სხვა მაგალითები მოიცავს USB შენახვის მოწყობილობების საჯაროდ ჩამოგდებას (მაგ. კომპანიის ავტოსადგომებში) იმ იმედით, რომ ვინმე (ხშირად კომპანიის თანამშრომელი) აიღებს ერთს და ჩააერთებს მას. მგრძნობიარე კომპიუტერი. ანალოგიურად, თავდამსხმელებმა შეიძლება დატოვონ "დამუხტვის კაბელი", რომელიც ჩართულია სოკეტში საჯარო ადგილას. სინამდვილეში, კაბელი შეიცავს მავნე პროგრამულ უზრუნველყოფას, როგორიცაა keyloggers ან ინსტრუმენტები მსხვერპლის მოწყობილობაზე კონტროლისთვის.

**Phishing types**

Different types of malicious emails can be classified as one of the following:

* [**Spam**](https://www.proofpoint.com/us/threat-reference/phishing) - unsolicited junk emails sent out in bulk to a large number of recipients. The more malicious variant of Spam is known as **MalSpam**.
* [**Phishing**](https://www.proofpoint.com/us/threat-reference/phishing)- emails sent to a target(s) purporting to be from a trusted entity to lure individuals into providing sensitive information.
* [**Spear phishing**](https://www.proofpoint.com/us/threat-reference/spear-phishing)**-**takes phishing a step further by targeting a specific individual(s) or organization seeking sensitive information.
* [**Whaling**](https://www.rapid7.com/fundamentals/whaling-phishing-attacks/)- is similar to spear phishing, but it's targeted specifically to C-Level high-position individuals (CEO, CFO, etc.), and the objective is the same.
* [**Smishing**](https://www.proofpoint.com/us/threat-reference/smishing) - takes phishing to mobile devices by targeting mobile users with specially crafted text messages.
* [**Vishing**](https://www.proofpoint.com/us/threat-reference/vishing) - is similar to smishing, but instead of using text messages for the social engineering attack, the attacks are based on voice calls.

**წვდომის კონტროლი - 10ქ.**

* ფიზიკური წვდომის კონტროლი;
* ლოგიკური წვდომის კონტროლი;
* ადმინისტრაციული წვდომის კონტროლი;
* სავალდებულო წვდომის კონტროლი, დისკრეციული წვდომის კონტროლი;
* როლზე დაფუძნებული კონტროლი, წესზე დაფუძნებული წვდომის კონტროლი;
* იდენტიფიკაცია, აუთენტიკაცია, ავტორიზაცია, ანგარიშგება;

**რა არის წვდომის კონტროლი?**

წვდომის კონტროლი არის უსაფრთხოების მექანიზმი, რომელიც გამოიყენება იმის გასაკონტროლებლად, რომელ მომხმარებლებს ან სისტემებს აქვთ უფლება შევიდნენ კონკრეტულ რესურსზე ან სისტემაზე. წვდომის კონტროლი დანერგილია კომპიუტერულ სისტემებში იმისთვის, რომ მხოლოდ ავტორიზებულ მომხმარებლებს ჰქონდეთ წვდომა რესურსებზე, როგორიცაა ფაილები, დირექტორიები, მონაცემთა ბაზები და ვებ გვერდები. წვდომის კონტროლის უპირველესი მიზანია დაცული იყოს სენსიტიური მონაცემები და უზრუნველყოს, რომ ისინი ხელმისაწვდომი იყოს მხოლოდ მათთვის, ვისაც აქვს მასზე წვდომის უფლება.

**დისკრეციული წვდომის კონტროლი (DAC):**

ამ ტიპის წვდომის კონტროლის დროს, რესურსის მფლობელი ან ადმინისტრატორი განსაზღვრავს, ვის აქვს რესურსზე წვდომის უფლება და რა ქმედებები აქვს მათ უფლებას შეასრულოს. DAC ჩვეულებრივ გამოიყენება ოპერაციულ სისტემებში და ფაილურ სისტემებში. ხალხური სიტყვებით რომ ვთქვათ, წარმოიდგინეთ ციხე, სადაც მეფეს შეუძლია მისცეს გასაღებები თავის მრჩევლებს, რაც მათ საშუალებას მისცემს გააღონ ნებისმიერი კარი, როცა მოესურვებათ. ეს არის DAC თქვენთვის. ეს არის თავისუფლება, გააკონტროლო წვდომა საკუთარ რესურსებზე. პასუხისმგებელს, ისევე როგორც ციხის მეფეს, შეუძლია ნებართვა გადასცეს, ვისაც სურს, კარნახოს, ვის შეუძლია შესვლა და გამოსვლა

**სავალდებულო წვდომის კონტროლი (MAC):**

ამ ტიპის წვდომის კონტროლის დროს რესურსებზე წვდომა განისაზღვრება წინასწარ განსაზღვრული წესების ან პოლიტიკის კომპლექტით, რომლებიც აღსრულებულია სისტემის მიერ. MAC ჩვეულებრივ გამოიყენება მაღალ უსაფრთხო გარემოში, როგორიცაა მთავრობა და სამხედრო სისტემები. ხალხური სიტყვებით რომ ვთქვათ, წარმოიდგინეთ ციხე რკინით დაფარული უსაფრთხოების პროტოკოლით. მხოლოდ კონკრეტულ პირებს, რომლებსაც აქვთ სპეციალური უსაფრთხოების ნებართვები, შეუძლიათ წვდომა გარკვეულ ადგილებში, და ეს არ არის შეთანხმებული. უმაღლესი მეთაური ადგენს წესებს და მათ მკაცრად იცავენ. ასე მუშაობს MAC. ეს ჰგავს უსაფრთხოების მკაცრ ოფიცერს, რომელიც არ უშვებს გამონაკლისს წესში.

**როლებზე დაფუძნებული წვდომის კონტროლი (RBAC):**

ამ ტიპის წვდომის კონტროლის დროს მომხმარებლებს ენიჭებათ როლები, რომლებიც განსაზღვრავენ რესურსებზე წვდომის დონეს. RBAC ჩვეულებრივ გამოიყენება საწარმოთა სისტემებში, სადაც მომხმარებლებს აქვთ უფლებამოსილების განსხვავებული დონე მათი სამუშაო პასუხისმგებლობის მიხედვით. ხალხური თვალსაზრისით, წარმოიდგინეთ თანამედროვე კორპორაცია. თქვენ გყავთ თქვენი მენეჯერები, თქვენი აღმასრულებლები, თქვენი გაყიდვების პერსონალი და ა.შ. თითოეულ მათგანს განსხვავებული წვდომა აქვს შენობაში. ზოგს შეუძლია შევიდეს გამგეობის დარბაზში, ზოგს შეუძლია გაყიდვების სართულზე შესვლა და ა.შ. ეს არის RBAC-ის არსი - წვდომის მინიჭება ორგანიზაციაში პიროვნების როლზე დაყრდნობით.

**ატრიბუტებზე დაფუძნებული წვდომის კონტროლი (ABAC):**

ამ ტიპის წვდომის კონტროლის დროს რესურსებზე წვდომა განისაზღვრება ატრიბუტების ნაკრებით, როგორიცაა მომხმარებლის როლი, დღის დრო, მდებარეობა და მოწყობილობა. ABAC ჩვეულებრივ გამოიყენება ღრუბლოვან გარემოში და ვებ აპლიკაციებში. ხალხური თვალსაზრისით, იფიქრეთ უაღრესად მოწინავე სამეცნიერო ფანტასტიკის უსაფრთხოების სისტემაზე, რომელიც სკანირებს ინდივიდებს გარკვეული ატრიბუტების დასადგენად. შესაძლოა ის ამოწმებს, არიან თუ არა ისინი კონკრეტული პლანეტიდან, ატარებენ თუ არა კონკრეტულ მოწყობილობას, ან ცდილობენ თუ არა რესურსზე წვდომას კონკრეტულ დროს. ეს არის ABAC. ეს მომავლის ჭკვიან, მოქნილ უსაფრთხოებას ჰგავს.

**ადმინისტრაციული წვდომის კონტროლი:**

განმარტება: ადმინისტრაციული წვდომის კონტროლი მოიცავს ორგანიზაციების მიერ დადგენილ პოლიტიკას, პროცედურებსა და მმართველობით ზომებს, რათა მართონ და გააკონტროლონ წვდომა როგორც ფიზიკურ, ასევე ციფრულ რესურსებზე. იგი მოიცავს წვდომის კონტროლის მექანიზმების ზოგად მართვას.

მაგალითები:

* წვდომის კონტროლის პოლიტიკა, რომელიც განსაზღვრავს მისაღებ გამოყენებას და წვდომის დონეებს.
* თანამშრომლების ბორტზე დაშვების პროცედურები.
* რეგულარული უსაფრთხოების ტრენინგები და ცნობიერების ამაღლების პროგრამები.
* წვდომის ჟურნალების აუდიტი და მონიტორინგი შესაბამისობისთვის.
* მიზანი: უზრუნველყოს ჩარჩო ეფექტური წვდომის კონტროლის განსახორციელებლად და შესანარჩუნებლად. ადმინისტრაციული წვდომის კონტროლი უზრუნველყოფს პოლიტიკის განსაზღვრას, კომუნიკაციას და აღსრულებას ორგანიზაციის მასშტაბით.

**ლოგიკური წვდომის კონტროლი:**

განმარტება: ლოგიკური წვდომის კონტროლი მოიცავს ტექნოლოგიის გამოყენებას ციფრულ სისტემებზე, ქსელებსა და მონაცემებზე წვდომის დასარეგულირებლად. ის ყურადღებას ამახვილებს მომხმარებლის ავტორიზაციის კონტროლზე და ინფორმაციულ სისტემებზე წვდომის ავტორიზაციაზე.

მაგალითები:

* მომხმარებლის სახელები და პაროლები სისტემაში შესვლისთვის.
* მრავალფაქტორიანი ავთენტიფიკაციის (MFA) მეთოდები.
* როლებზე დაფუძნებული წვდომის კონტროლი (RBAC) როლების მიხედვით ნებართვების მინიჭება.
* წვდომის კონტროლის სიები (ACL) ფაილების ან ქსელის რესურსების ნებართვების მითითებით.
* მიზანი: ციფრული აქტივების, მონაცემებისა და საინფორმაციო სისტემების დაცვა იმით, რომ მხოლოდ ავტორიზებულ პირებს შეუძლიათ მათთან წვდომა, შეცვლა ან ურთიერთქმედება.

**ფიზიკური წვდომის კონტროლი:**

განმარტება: ფიზიკური წვდომის კონტროლი ეხება ზომებს და მექანიზმებს, რომლებიც გამოიყენება ფიზიკურ სივრცეებზე, ობიექტებსა და აქტივებზე წვდომის შესაზღუდად ან მისაცემად, როგორიცაა შენობები, ოთახები ან მონაცემთა ცენტრები.

მაგალითები:

* ფიზიკური ბარიერები, როგორიცაა ღობეები, კედლები და კარიბჭე.
* საკეტები, გასაღები ბარათები, ბიომეტრიული სკანერები ან PIN კოდები კარებზე.
* დაცვის თანამშრომლები ან სათვალთვალო სისტემები, რომლებიც აკონტროლებენ შესასვლელებს.
* ვიზიტორთა მართვის სისტემები სტუმრების წვდომის თვალყურის დევნისა და კონტროლისთვის.
* მიზანი: არაავტორიზებული პირების ფიზიკურად შეღწევის ან შეზღუდულ ადგილებში ურთიერთობის აღკვეთა, აქტივებისა და პერსონალის ფიზიკური უსაფრთხოების უზრუნველყოფა.

**Just in case**

1. იდენტიფიკაცია: identification

განმარტება: იდენტიფიკაცია არის უნიკალური იდენტიფიკატორის (როგორიცაა მომხმარებლის სახელი, თანამშრომლის ID, ან ანგარიშის ნომერი) წარდგენის პროცესი სისტემაში ან ერთეულზე საკუთარი ვინაობის დეკლარაციის მიზნით.

მაგალითი: მომხმარებლის სახელის შეყვანა შესვლის ეკრანზე.

მიზანი: დადგინდეს საწყისი პრეტენზია იმის შესახებ, თუ ვინ არის მომხმარებელი.

2. ავთენტიფიკაცია: authentification

განმარტება: ავტორიზაცია არის მოთხოვნილი პირადობის გადამოწმების პროცესი რწმუნებათა სიგელების გამოყენებით (მაგ., პაროლები, ბიომეტრია, უსაფრთხოების ნიშნები), რათა დარწმუნდეს, რომ მომხმარებელი არის ის, ვინც ამბობს, რომ არის.

მაგალითი: პაროლის შეყვანა ან თითის ანაბეჭდის მიწოდება ბიომეტრიული ავთენტიფიკაციისთვის.

მიზანი: მომხმარებლის მოთხოვნილი პირადობის ლეგიტიმურობის დადასტურება.

3. ავტორიზაცია: authorization

განმარტება: ავტორიზაცია არის მომხმარებლის ან სისტემისთვის წვდომის უფლებებისა და ნებართვების მინიჭების ან უარის თქმის პროცესი მათი დამოწმებული იდენტობისა და წვდომის დონის მიხედვით, რომელიც მათ უნდა ჰქონდეთ.

მაგალითი: მომხმარებლისთვის კონკრეტული როლების ან ნებართვების მინიჭება შესვლის შემდეგ, იმის დადგენა, თუ რა ქმედებები აქვთ უფლება შეასრულონ.

დანიშნულება: რესურსებზე წვდომის კონტროლი და შეზღუდვა ავტორიზებული მომხმარებლის პრივილეგიებზე დაყრდნობით.

4. მოხსენება: reporting

განმარტება: მოხსენება მოიცავს წვდომის მოვლენებისა და აქტივობების აღრიცხვას და მონიტორინგს. იგი მოიცავს შესვლის წარმატებული და წარუმატებელი მცდელობების ჩაწერას, სენსიტიურ მონაცემებზე წვდომას და უსაფრთხოების საკითხებთან დაკავშირებულ სხვა მოვლენებს.

მაგალითი: აუდიტის ჟურნალების გენერირება, რომელიც აღწერს შესვლის მცდელობებს, წვდომის ნებართვებში ცვლილებებს ან უსაფრთხოების ინციდენტს.

მიზანი: მომხმარებლის საქმიანობის ჩანაწერის უზრუნველყოფა აუდიტის, შესაბამისობისა და უსაფრთხოების ინციდენტებზე რეაგირებისთვის. ანგარიშები დაგეხმარებათ საეჭვო ქცევის იდენტიფიცირებაში, ცვლილებების თვალყურის დევნებაში და ანგარიშვალდებულების შენარჩუნებაში.

**კიბერუსაფრთხოების როლები - 10ქ.**

1. ინფორმაციული უსაფრთხოების მთავარი ოფიცერი (CISO): Chief Information Security Officer (CISO):

მიმოხილვა: CISO არის უმაღლესი აღმასრულებელი თანამდებობის პირი, რომელიც პასუხისმგებელია ორგანიზაციის შიგნით ინფორმაციის უსაფრთხოების მთლიანი სტრატეგიისა და პოლიტიკის ზედამხედველობაზე და მართვაზე.

პასუხისმგებლობები: შეიმუშავებს და ახორციელებს უსაფრთხოების პოლიტიკას, მართავს რისკებს, უზრუნველყოფს შესაბამისობას და დაუკავშირდება სხვა აღმასრულებლებსა და დაინტერესებულ მხარეებს ორგანიზაციის უსაფრთხოების მდგომარეობის შესახებ.

1. შეღწევადობის ტესტერი (ეთიკური ჰაკერი):Penetration Tester (Ethical Hacker):

მიმოხილვა: ახორციელებს ავტორიზებულ სიმულაციურ კიბერშეტევებს სისტემებსა და ქსელებში დაუცველობის დასადგენად.

პასუხისმგებლობები: ახორციელებს შეღწევადობის ტესტირებას, დაუცველობის შეფასებას და უზრუნველყოფს რეკომენდაციებს უსაფრთხოების პოზის გასაუმჯობესებლად. მუშაობს ეთიკურ ჰაკერზე პოტენციური სისუსტეების გამოსავლენად და მოსაგვარებლად.

1. ინციდენტის რეაგირება:Incident Responder:

მიმოხილვა: სპეციალიზირებულია უსაფრთხოების ინციდენტების მართვასა და შერბილებაში, როგორიცაა მონაცემთა გარღვევა ან კიბერშეტევები.

პასუხისმგებლობები: შეიმუშავებს და ახორციელებს ინციდენტებზე რეაგირების გეგმებს, იძიებს უსაფრთხოების ინციდენტებს და კოორდინაციას უწევს სხვადასხვა გუნდებს საფრთხეების შეკავებისა და აღმოფხვრის მიზნით.

1. უსაფრთხოების არქიტექტორი:Security Architect:

მიმოხილვა: შეიმუშავებს და ახორციელებს უსაფრთხო საინფორმაციო სისტემებს უსაფრთხოების მოთხოვნების და დაგეგმვის არქიტექტურის განსაზღვრით.

პასუხისმგებლობები: შეიმუშავებს უსაფრთხოების ჩარჩოებს, უზრუნველყოფს უსაფრთხოების ინტეგრაციას სისტემებში და აფასებს აპლიკაციებისა და ქსელების უსაფრთხოების დიზაინს.

1. კიბერ, იურიდიული, პოლიტიკისა და შესაბამისობის ოფიცერი:Cyber, Legal, Policy, and Compliance Officer:

მიმოხილვა: ყურადღებას ამახვილებს კიბერუსაფრთხოების სამართლებრივ და მარეგულირებელ ასპექტებზე, კანონებთან და პოლიტიკასთან შესაბამისობის უზრუნველყოფაზე.

პასუხისმგებლობები: შეიმუშავებს და ახორციელებს კიბერუსაფრთხოების პოლიტიკას, უზრუნველყოფს კანონის შესაბამისობას და მართავს მარეგულირებელ მოთხოვნებს.

1. კიბერ საფრთხეების დაზვერვის სპეციალისტი:Cyber Threat Intelligence Specialist:

მიმოხილვა: აგროვებს და აანალიზებს ინფორმაციას პოტენციური და მიმდინარე კიბერ საფრთხეების შესახებ გადაწყვეტილების მიღებისა და უსაფრთხოების ზომების გასაძლიერებლად.

პასუხისმგებლობები: მონიტორინგს უწევს საფრთხის ლანდშაფტებს, ატარებს სადაზვერვო ანალიზს და აწვდის ქმედით ინფორმაციას კიბერუსაფრთხოების დაცვის გასაძლიერებლად.

1. კიბერუსაფრთხოების აუდიტორი:Cybersecurity Auditor:

მიმოხილვა: იკვლევს და აფასებს ორგანიზაციის საინფორმაციო სისტემებს უსაფრთხოების პოლიტიკასთან და სტანდარტებთან შესაბამისობის უზრუნველსაყოფად.

მოვალეობები: ატარებს აუდიტს, აფასებს უსაფრთხოების კონტროლს და უზრუნველყოფს რეკომენდაციებს უსაფრთხოების პრაქტიკის გასაუმჯობესებლად.

1. კიბერუსაფრთხოების განმანათლებელი:Cybersecurity Educator:

მიმოხილვა: ფოკუსირებულია ორგანიზაციის ფარგლებში ინდივიდების განათლებაზე კიბერუსაფრთხოების საუკეთესო პრაქტიკისა და პოლიტიკის შესახებ.

მოვალეობები: შეიმუშავებს სასწავლო პროგრამებს, ატარებს ცნობიერების ამაღლების კამპანიებს და უზრუნველყოფს თანამშრომლების კარგად ინფორმირებას კიბერუსაფრთხოების რისკებისა და პრევენციული ზომების შესახებ.

1. კიბერუსაფრთხოების განმახორციელებელი: Cybersecurity Implementer:

მიმოხილვა: ახორციელებს და მართავს კიბერუსაფრთხოების ტექნოლოგიებსა და გადაწყვეტილებებს ორგანიზაციაში.

პასუხისმგებლობები: ახორციელებს უსაფრთხოების ინსტრუმენტებს, აკონფიგურირებს სისტემებს და უზრუნველყოფს უსაფრთხოების ზომების ეფექტურ მოქმედებას.

1. კიბერუსაფრთხოების მკვლევარი: Cybersecurity Researcher:

მიმოხილვა: ატარებს კვლევას კიბერუსაფრთხოების სფეროში ახალი და ახალი საფრთხეების, მოწყვლადობისა და ტექნოლოგიების გამოსავლენად.

პასუხისმგებლობები: აკვირდება ინდუსტრიის განვითარებას, ხელს უწყობს ცოდნის გაზიარებას და იკვლევს ინოვაციურ გადაწყვეტილებებს კიბერუსაფრთხოების გასაძლიერებლად.

1. კიბერუსაფრთხოების რისკის მენეჯერი: Cybersecurity Risk Manager:

მიმოხილვა: მართავს და აფასებს კიბერუსაფრთხოების რისკებს ორგანიზაციის შიგნით, იმის უზრუნველსაყოფად, რომ რისკები გასაგებია და სათანადოდ შერბილებული.

პასუხისმგებლობები: ატარებს რისკების შეფასებას, შეიმუშავებს რისკების მართვის სტრატეგიებს და მუშაობს კიბერუსაფრთხოების საფრთხეების პოტენციური ზემოქმედების მინიმუმამდე შემცირებაზე.

1. ციფრული სასამართლო ექსპერტიზის გამომძიებელი: Digital Forensics Investigator:

მიმოხილვა: სპეციალიზირებულია ელექტრონული მტკიცებულებების შეგროვებაში, ანალიზსა და შენარჩუნებაში, რათა გამოიძიოს და რეაგირება მოახდინოს კიბერ ინციდენტებზე.

პასუხისმგებლობები: ატარებს ციფრულ სასამართლო ექსპერტიზას, აგროვებს მტკიცებულებებს სასამართლო პროცესისთვის და ეხმარება ინციდენტზე რეაგირების გამოძიებაში.

**საბოლოო მოწყობილობები - 10ქ**

საბოლოო მოწყობილობა არის, მოწყობილობა ან გარკვეული ლოკაცია ქსელში სადაც კომუნიკაცია იწყება ან მთავრდება, ეს შეიძლება იყოს ფიზიკური დივაისი, როგორებიცაა კომპიუტერი, ლეპტოპი სერვერი, ან ვირტუალური ერთეული მაგალითად ვირტუალური მანქანა არ ქსელის სერვისი. ენდფოინთები ქსელის საჭირო კომპონენტებია, რადგან ისინი უზრუნველყოფენ კომუნიკაციას იუზერებს, აპლიკაციებს ან დეივაისებს შორის.მათ შეუძლიათ გაგზავნონ ან მიიღონ დატა, მიიღონ წვდომა ქსელურ რესურსებზე ან ირუთიერთქმედონ სხვა ენდფოინთებთან ან სერვისებთან.

რაც შეეხება სისუსტეებს, ენდფოით დივაისებს როგორორებიცაა, კომპიუტერი, ლეპტოპი, სმარტფონი და ტაბლეტი შეიძლება გააჩნდეს სისუსტეები მათ დაცვის კონტროლზე, რაც გამოიწვევს მათ დაუცველობას შეტევის მიმართ. მათ შორის გაცრცელებული სისუსტეებია:

ვადაგასული სოფთვეარი და ოპერაციული სისტემა: ენდფოინთებს ვადაგასული სოფთვეართით ან ოპერაციული სისტემით შეიძლება ჰქონდეს ეთექერებისთვის ცნობილი დაუცველობები რომლებსაც შემდგომ ისინი გამოიყენებენ.

სუსტი პაროლები: ენდფოინთები სუსტი ან მარტივად გამოსაცნობი პაროლებით შეიძლება გახდეს ბრიუთფორს შეტევის ან პაროლის გამოცნობის შეტევის მსხვერპლი.

დაცვის სოფთვეარის არარსებობა: ედნფოინთები ანტივირუსის ან ანტიმალვეარის ან სხვა რაიმე დაცვის გარეშე, ხდება დაუცველი და მარტივად ხელმისაწვდომი ათაქერებისთვის.

არასწორად დაკონფიგურირებული დაცვის კონტროლის მექანიზმები როგორიცაა მაგალითად ფაირვოლი შეიძლება ენდფოინთის დაუცველობის მთავარი მიზეზი გახდეს.

არადაცული დისტანციური წვდომა როგორიცაა სუსტი პაროლები ან დაუშიფრავი კავშირი ასევე შესაძლებელია დაუცველობის მთავარი მიზეზი გახდეს

სოციალური ინჟინერიის შეტევები ენდფოინთის კიდევ ერთი სისუსტეა, როგორიცაა ფიშინი ან სფეარ ფიშინგი

ფიზიკური ქურდობა ან დაკარგვა: ენდფოინთები შეიძლება მოიპარონ ან დაიკარგოს რაც გამოიწვევს სენსიტიური ინფორმაციის გაჟონვას ან ათაქერებს უფლებას მისცემს წვდომა მოიპოვონ შიდა სისტემაზე.

ენდფოინთების კიდევ ერთი სისუსტე შეიძლება იყოს მომხარებლის შეცდომა რომელიც მგალითად გადმოწერს რაიმე მალვეარს.

ორგანიზაციებისთვის მნიშვენლოვანია რომ რეგულარულად აკონტროლოს თავიანთი ენდფოინთების დაცულობა რომ მინიმალურ რისკებამდე დაიყვანონ რაიმე ჩავარდნა დაცვაში.ეს შეიძლება იყოს, ძლიერი პაროლები,

სოფთევარის და სისტემის რეგულარული აპდეითი, გამართული დაცვის სისტემა და ა.შ

........................................................

**ციფრული სერთიფიკატები -** ციფრული სერთიფიკატი არის ციფრული ფაილი, რომელიც ადასტურებს ინდივიდის, ორგანიზაციის ან ვებსაიტის იდენტურობას. ის შეიცავს ინფორმაციას, როგორიცაა მფლობელის სახელი, საჯარო გასაღები და სხვა საიდენტიფიკაციო ინფორმაცია, ასევე ციფრული ხელმოწერა სანდო მესამე მხარისგან, რომელიც ცნობილია როგორც სერტიფიკატის ავტორიტეტი.

**ბლოკური და ნაკადური შიფრები -** ნაკადური შიფრაციის დროს დატას თითოეული ბაიტი ან ბიტი იშიფრება ერთდროულად, ამ ტიპის შიფრაცია კარგია იმ ტიპის კომუნიკაციის დროს როდესაც დატას ან მესიჯის სრული სიგრძე არ არის ცნობილი.

რაც შეეხება ბლოკურ შფრაცას დატა იყოფა ტოლი სიგრძის ბლოკებად

ძირითადად 128ბიტის ტოლია, თუ ზუსტად არ იყოფა დატა მაშინ მას ემატება საჭირო ბიტების რაოდენობა რაიმე სტრინგით ალგორითმის გათვალისწინებით.

**კრიპტოგრაფიული გასაღები -** კრიპტოგრაფიული გასაღები, არის ინფორმაციის ნაწილი, რომელიც გამოიყენება კრიპტოგრაფიულ ალგორითმებში მონაცემთა დაშიფვრის ან გაშიფვრის მიზნით. ეს არსებითად არის ბიტების ან სიმბოლოების სტრიქონი, რომელიც ემსახურება როგორც პარამეტრს, რათა აკონტროლოს ჩვეულებრივი ტექსტის ტრანსფორმაცია შიფრულ ტექსტად ან პირიქით. კრიპტო გასაღებები აუცილებელია კრიპტოგრაფიული სისტემების მონაცემების კონფიდენციალურობის, მთლიანობისა და ავთენტურობის უზრუნველსაყოფად.

**ციფრული ხელმოწერა -** ციფრული ხელმოწერები არის ფაილების ავთენტურობის დასამტკიცებლად, იმის დასამტკიცებლად, თუ ვინ შექმნა ან შეცვალა ისინი. ასიმეტრიული კრიპტოგრაფიის გამოყენებით, თქვენ აწარმოებთ ხელმოწერას თქვენი პირადი გასაღებით და მისი გადამოწმება შესაძლებელია თქვენი საჯარო გასაღების გამოყენებით. ვინაიდან მხოლოდ თქვენ უნდა გქონდეთ წვდომა თქვენს პირად გასაღებზე, ეს ადასტურებს, რომ თქვენ მოაწერეთ ხელი ფაილს.

**ქეში -** თითოეულ მოწყობილობას ქსელში აქვს ჩანაწერი ინფორმაციის შესანახად, რომელსაც ქეში ეწოდება.  ARP პროტოკოლის კონტექსტში, ეს ქეში ინახავს ქსელში არსებული სხვა მოწყობილობების იდენტიფიკატორებს.

**Salting -** Salting არის ტექნიკა, რომელიც გამოიყენება კრიპტოგრაფიაში პაროლის შენახვისა და კრიპტოგრაფიული ჰეშის ფუნქციების უსაფრთხოების გასაძლიერებლად. იგი გულისხმობს შემთხვევითი ან უნიკალური მნიშვნელობის დამატებას, რომელიც ცნობილია როგორც salt, შეყვანისთვის, სანამ ის ჰეშირდება ან დაშიფრული იქნება. დანამატი, როგორც წესი, არის ბიტების ან სიმბოლოების შემთხვევითი სტრიქონი, რომელიც გენერირებულია თითოეული მომხმარებლისთვის ან მონაცემთა შეყვანისთვის.

**სიმეტრიული და ასიმეტრიული დაშიფვრა -** სიმეტრიული დაშიფვრა იყენებს ერთ სექრეთ ქის როგოც დაშიფვრისთვის ასევე გაშიფვრისთვის. სიმეტრიული დაშიფვრა ძალიან სწრაფია და გამოიყენება დიდი მოცულობის მონაცემების დასაშფვრად.

რაც შეეხება ასიმეტრიულ დაშფვრას, ის იყენებს ორ სხვადასხვა ქის დასაშფრად და გასაშიფრად ერთი გასაღები არის საჯარო და შესაძლებელია ყველას გავუზიაროთ ხოლო მეროე გასაღები არის დაცული და მხოლოდ მფლობელმა იცის.საჯარო გასაღებით დაშიფვრული მონაცემები მხოლოდ ამ დაცული გასაღებით შეიძლება გაიშიფროს და პირიქით.

**ცისარტყელას ცხრილი -** ცისარტყელას ცხრილი არის წინასწარ გამოთვლილი ცხრილი, რომელიც შეიცავს უამრავ პოტენციურ უბრალო ტექსტს ჰეშის ფუნქციებისთვის. ცისარტყელას ცხრილის დანიშნულებაა დააჩქაროს ჰეშირებული მნიშვნელობების თავდაპირველი ტექსტის ფორმაში დაბრუნება, პაროლების ან სხვა ჰეშირებული მონაცემების ეფექტურად გატეხვა.

**იდენთიფიკაცია ავთენტიპიკაცია ავტორიზაცია** - იდენთიფიკაცია: ანგარიშის ან პირადობის მოწმობის შექმნა, რომელიც ცალსახად

წარმოადგენს მომხმარებელს, მოწყობილობას ან პროცესს ქსელში.

ავთენტიფიკაცია: იმის დამტკიცება, რომ საგანი არის ის ვინც ან რაც აცხადებს რომ ცდილობს რესურსზე წვდომას.

ავტორიზაცია: მოიცავს, თუ რა უფლებები უნდა ჰქონდეთ სუბიექტებს თითოეულ

რესურსზე და ამ უფლებების აღსრულებას.

• იდენტიფიკაცია - დარწმუნდით, რომ მომხმარებლები არიან ლეგიტიმური. მაგალითად, შეიძლება დაგჭირდეთ დარწმუნდეთ, რომ ბილინგისა და მიწოდების მისამართები ემთხვევა და რომ ისინი არ ცდილობენ გამოიყენონ თაღლითური გადახდის მეთოდები.

• ავთენტიფიკაცია - დარწმუნდით, რომ კლიენტებს აქვთ უნიკალური ანგარიშები და რომ მხოლოდ მათ შეუძლიათ თავიანთი შეკვეთების და ბილინგის ინფორმაციის მართვა.

• ავტორიზაცია - წესები, რომლებიც უზრუნველყოფენ კლიენტებს შეკვეთების განთავსება მხოლოდ მაშინ, როდესაც მათ აქვთ გადახდის მოქმედი მექანიზმები. თქვენ შეიძლება აწარმოოთ ლოიალობის სქემები ან აქციები, რომლებიც უფლებას აძლევს გარკვეულ მომხმარებელს ნახონ უნიკალური შეთავაზებები ან კონტენტი.

**ინფორმაციულობა მთლიანობა განსაზღვრა -** ინფორმაციის უსაფრთხოება (ან infosec) მოიცავს მონაცემთა რესურსების დაცვას არაავტორიზებული წვდომისგან, თავდასხმისგან, ქურდობისგან ან დაზიანებისგან. მონაცემები შეიძლება იყოს დაუცველი მათი შენახვის, გადაცემის ან პროცესში. ინფორმაციული უსაფრთხოების სამი ძირითადი პრინციპი, რომელსაც მოიხსენიებენ როგორც CIA (confidentiality integrity availability) ტრიადად, არის:

კონფიდენციალურობა ნიშნავს, რომ გარკვეული ინფორმაცია უნდა იყოს ცნობილი მხოლოდ გარკვეული ადამიანებისთვის.

მთლიანობა ნიშნავს, რომ მონაცემები ინახება და გადაცემულია ისე, როგორც დანიშნულებისამებრ, და რომ ნებისმიერი ცვლილება ავტორიზებულია.

ხელმისაწვდომობა ნიშნავს, რომ ინფორმაცია ხელმისაწვდომია მათთვის, ვინც უფლებამოსილია ნახოს ან შეცვალოს იგი.

**მალვეარი -** მალვეარი არის მავნე პროგრამა რომელსაც დიდი ზიანის გამოწვევა შეუძლია როგორც კომპიუტერული სისტემისთვის ისევე მომხარმებლისთივს. მალვეარის ბევრი ტიპი არსებობს ესენია : ვირუსები, ტროას ცხენები, ჭიები ანუ ვორმები, რენსომვეარი, ბექდორი და რუთკიტი.

**ინციდენტზე რეაგირება -** ინციდენტებზე რეაგირების პოლიტიკა ადგენს რესურსებს, პროცესებსა და მითითებებს უსაფრთხოების ინციდენტების მოსაგვარებლად. ინციდენტების მართვა სასიცოცხლოდ მნიშვნელოვანია რისკის შესამცირებლად. ის მოიცავს მომზადებას, იდენტიფიცირებას, შეკავებას, აღმოფხვრას, აღდგენას, გაკვეთილის სწავლას.

**რისკი -** რისკი ეხება სენსიტიურ ინფორმაციაზე, სისტემებსა თუ რესურსებზე დარღვევის, კომპრომისის ან არაავტორიზებული წვდომის პოტენციალს.

რისკის იდენტიფიკაცია არის პოტენციური რისკების ან საფრთხეების იდენტიფიცირებისა და გაგების პროცესი, რომელთა წინაშეც შეიძლება აღმოჩნდეს ორგანიზაცია. ის გულისხმობს სხვადასხვა ფაქტორების სისტემატურ იდენტიფიცირებას, დოკუმენტირებას და ანალიზს, რამაც შეიძლება გამოიწვიოს უსაფრთხოების ინციდენტები ან დაუცველობა. რისკის იდენტიფიკაციის მიზანია ორგანიზაციის პოტენციური რისკების გააზრება, რათა შეიმუშაოს რისკის მართვის ეფექტური სტრატეგიები და კონტროლი.

მისიის არსებითი ფუნქციების იდენტიფიცირება

მოწყვლადობის იდენტიფიცირება

საფრთხეების იდენტიფიცირება

გაანალიზეთ ბიზნესის გავლენა

რისკზე რეაგირების იდენტიფიცირება

**ფაირვოლი და ანტივირუსი:** Firewall არის მოწყობილობა ქსელში, რომელიც პასუხისმგებელია იმის განსაზღვრაზე, თუ რა ტრაფიკს შეუძლია შესვლა და გასვლა. იფიქრეთ firewall- ზე, როგორც ქსელის საზღვრის უსაფრთხოებაზე. ადმინისტრატორს შეუძლია დააკონფიგურიროს firewall, რათა დაუშვას ან უარყოს ტრაფიკი ქსელში შესვლის ან გასვლისგან, მრავალი ფაქტორის საფუძველზე, როგორიცაა:

საიდან მოდის ტრაფიქი

სად მიდის ტრაფიქი

რომელი პორტისთვისაა ტრაფიქი

რა პროტოკოლს იყენებს ტრაფიკი

ანტივირუსული (AV) პროგრამა არის უსაფრთხოების დამატებითი ფენა, რომელიც მიზნად ისახავს აღმოაჩინოს და თავიდან აიცილოს მავნე ფაილების გაშვება და გავრცელება სამიზნე ოპერაციულ სისტემაში.

**კონფიდენციალურობა:**

**მარტივი ახსნა: იფიქრეთ კონფიდენციალურობაზე, როგორც საიდუმლო კოდი. საუბარია იმაზე, რომ დარწმუნდეთ, რომ მხოლოდ ადამიანებმა უნდა იცოდნენ ეს. მაგალითად, თუ პაროლი გაქვთ, არ გინდათ, რომ სხვებმა იცოდნენ, თუ არ იტყვით, რომ ეს ნორმალურია.**

**მთლიანობა:**

**მარტივი ახსნა: მთლიანობა იგივეა, რომ დარწმუნდეთ, რომ თქვენი საყვარელი თამაში ან ამბავი იგივე დარჩება და არ აერევა. თქვენ არ გსურთ ვინმემ შეცვალოს თქვენი თამაშის წესები ან თქვენი საყვარელი ისტორიის სიტყვები დაუკითხავად, არა?**

**ხელმისაწვდომობა:**

**მარტივი ახსნა: ხელმისაწვდომობა არის ის, რომ დარწმუნდეთ, რომ თქვენთვის საჭირო ნივთები იქ არის, როცა გინდათ. ისევე, როგორც თქვენ ელით, რომ თქვენი საყვარელი სათამაშო ან თამაში მზად იქნება სათამაშოდ, როცა მოგეწონებათ.**

**Antivirus**

What is AV software?

Antivirus (AV) software is an extra layer of security that aims to detect and prevent the execution and spread of malicious files in a target operating system.

It is a host-based application that runs in real-time (in the background) to monitor and check the current and newly downloaded files. The AV software inspects and decides whether files are malicious using different techniques.

Interestingly, the first antivirus software was designed solely to detect and remove computer viruses. Nowadays, that has changed; modern antivirus applications can detect and remove computer viruses as well other harmful files and threats.

**What does AV software look for?**

Traditional AV software looks for **malware** with predefined malicious patterns or signatures. Malware is harmful software whose primary goal is to cause damage to a target machine, including but not limited to:

* Gain full access to a target machine.
* Steal sensitive information such as passwords.
* Encrypt files and cause damage to files.
* Inject other malicious software or unwanted advertisements.
* Used the compromised machine to perform further attacks such as botnet attacks.
* **AV vs other security products**
* In addition to AV software, other host-based security solutions provide real-time protection to endpoint devices. Endpoint Detection and Response (EDR) is a security solution that provides real-time protection based on behavioral analytics. An antivirus application performs scanning, detecting, and removing malicious files. On the other hand, EDR monitors various security checks in the target machine, including file activities, memory, network connections, Windows registry, processes, etc.
* Modern Antivirus products are implemented to integrate the traditional Antivirus features and other advanced functionalities (similar to EDR functionalities) into one product to provide comprehensive protection against digital threats.

**Antivirus Engines**

An AV engine is responsible for finding and removing malicious code and files. Good AV software implements an effective and solid AV core that accurately and quickly analyzes malicious files. Also, It should handle and support various file types, including archive files, where it can self-extract and inspect all compressed files.

Most AV products share the same common features but are implemented  differently, including but not limited to:

* Scanner
* Detection techniques
* Compressors and Archives
* Unpackers
* Emulators
* **Scanner**
* The scanner feature is included in most AV products: AV software runs and scans in real-time or on-demand. This feature is available in the GUI or through the command prompt. The user can use it whenever required to check files or directories. The scanning feature must support the most known malicious file types to detect and remove the threat. In addition, it also may support other types of scanning depending on the AV software, including vulnerabilities, emails, Windows memory, and Windows Registry.

**Detection techniques**

An AV detection technique searches for and detects malicious files; different detection techniques can be used within the AV engine, including:

* Signature-based detection is the traditional AV technique that looks for predefined malicious patterns and signatures within files.
* Heuristic detection is a more advanced technique that includes various behavioral methods to analyze suspicious files.
* Dynamic detection is a technique that includes monitoring the system calls and APIs and testing and analyzing in an isolated environment.

We will cover these techniques in the next task. A good AV engine is accurate and quickly detects malicious files with fewer false-positive results.

**Compressors and Archives**

The "Compressors and Archives" feature should be included in any AV software. It must support and be able to deal with various system file types, including compressed or archived files: ZIP, TGZ, 7z, XAR, RAR, etc. Malicious code often tries to evade host-based security solutions by hiding in compressed files. For this reason, AV software must decompress and scan through all files before a user opens a file within the archive.

**PE (Portable Executable) Parsing and Unpackers**

Malware hides and packs its malicious code by compressing and encrypting it within a payload. It decompresses and decrypts itself during runtime to make it harder to perform static analysis. Thus, AV software must be able to detect and unpack most of the known packers (UPX, Armadillo, ASPack, etc.) before the runtime for static analysis.

Malware developers use various techniques, such as Packing, to shrink the size and change the malicious file's structure. Packing compresses the original executable file to make it harder to analyze. Therefore, AV software must have an unpacker feature to unpack protected or compressed executable files into the original code.

Another feature that AV software must have is Windows Portable Executable (PE) header parser. Parsing PE of executable files helps distinguish malicious and legitimate software (.exe files). The PE file format in Windows (32 and 64 bits) contains various information and resources, such as object code, DLLs, icon files, font files, and core dumps.

**Emulators**

An emulator is an Antivirus feature that does further analysis on suspicious files. Once an emulator receives a request, the emulator runs the suspect (exe, DLL, PDF, etc.) files in a virtualized and controlled environment. It monitors the executable files' behavior during the execution, including the Windows APIs calls, Registry, and other Windows files. The following are examples of the artifacts that the emulator may collect:

* API calls
* Memory dumps
* Filesystem modifications
* Log events
* Running processes
* Web requests

An emulator stops the execution of a file when enough artifacts are collected to detect malware.

**Other common features**

The following are some common features found in AV products:

* A self-protection driver to guard against malware attacking the actual AV.
* Firewall and network inspection functionality.
* Command-line and graphical interface tools.
* A daemon or service.
* A management console.

**Logging**

Monitoring common protocols is essential for maintaining the security, performance, and reliability of a network. Here's a brief overview of monitoring considerations for some common protocols:

1. **Syslog:**
   1. **Monitoring Points:** Collect syslog messages from network devices, servers, and applications.
   2. **Tools:** Syslog servers, such as syslog-ng, rsyslog, or Splunk, can be used to collect, store, and analyze syslog data.
   3. **Metrics to Monitor:** Message volume, severity levels, source IPs, and specific message patterns.
2. **NTP (Network Time Protocol):**
   1. **Monitoring Points:** Monitor time synchronization between NTP servers and clients.
   2. **Tools:** NTP monitoring tools like ntpq or dedicated NTP monitoring solutions.
   3. **Metrics to Monitor:** Time offset, jitter, reachability, and stratum levels.
3. **DNS (Domain Name System):**
   1. **Monitoring Points:** Monitor DNS resolution times and DNS query volumes.
   2. **Tools:** DNS monitoring tools like dnstop, dnsmasq, or dedicated DNS monitoring solutions.
   3. **Metrics to Monitor:** Query response times, query success rates, and authoritative server responsiveness.
4. **HTTP and HTTPS:**
   1. **Monitoring Points:** Monitor web servers, load balancers, and proxies.
   2. **Tools:** Web server logs, application performance monitoring (APM) tools, or HTTP/HTTPS-specific monitoring tools.
   3. **Metrics to Monitor:** Response times, error rates, HTTP status codes, and server resource utilization.
5. **ICMP (Internet Control Message Protocol):**
   1. **Monitoring Points:** Monitor network connectivity and latency.
   2. **Tools:** Ping, traceroute, or network monitoring tools like Nagios, Zabbix, or PRTG.
   3. **Metrics to Monitor:** Round-trip time, packet loss, and route changes.
6. **Mail Protocols (e.g., SMTP, POP3, IMAP):**
   1. **Monitoring Points:** Monitor mail servers and associated protocols.
   2. **Tools:** Mail server logs, network monitoring tools, or specialized email monitoring solutions.
   3. **Metrics to Monitor:** Email delivery times, server response times, and mail queue lengths.
7. It's crucial to set up alerts based on predefined thresholds for these metrics to proactively identify and address issues. Additionally, consider using centralized monitoring solutions that can aggregate data from different sources and provide a holistic view of network health. Regularly review and update monitoring configurations to adapt to changes in network infrastructure and application environments.I

Log files are crucial sources of information for monitoring and analyzing the security and performance of IT environments. Different types of logs provide insights into various aspects of system and network activities. Here are some common security data types found in log files, along with examples of end device logs and network logs:

**End Device Logs:**

1. **System Logs:**
   1. **Description:** General system events, errors, and warnings.
   2. **Examples:** Windows Event Logs (System), /var/log/syslog (Linux).
2. **Application Logs:**
   1. **Description:** Records events specific to applications running on the device.
   2. **Examples:** Application-specific log files.
3. **Device Management Logs:**
   1. **Description:** Captures events related to device configuration changes and management activities.
   2. **Examples:** Router or switch logs.
4. **User Activity Logs:**
   1. **Description:** Records user actions on the device.
   2. **Examples:** Bash history (Linux), PowerShell logs (Windows).

**Security Data Types:**

1. **Authentication Logs:**
   1. **Description:** Records information about user logins and authentication attempts.
   2. **Examples:** Windows Event Logs (Security), /var/log/auth.log (Linux).
2. **Authorization Logs:**
   1. **Description:** Tracks user permissions and access control decisions.
   2. **Examples:** Windows Event Logs (Security), /var/log/auth.log (Linux).
3. **Audit Logs:**
   1. **Description:** Captures detailed records of system and application activities for security analysis.
   2. **Examples:** Windows Security Log, /var/log/audit/audit.log (Linux).
4. **Intrusion Detection and Prevention System (IDPS) Logs:**
   1. **Description:** Records alerts and events related to potential security threats.
   2. **Examples:** Snort logs, Suricata logs.
5. **Firewall Logs:**
   1. **Description:** Logs information about traffic blocked or allowed by the firewall.
   2. **Examples:** iptables logs (Linux), Windows Firewall logs.
6. **Antivirus and Anti-malware Logs:**
   1. **Description:** Provides information on detected or blocked malicious activities.
   2. **Examples:** Logs from antivirus software.
7. **Endpoint Security Logs:**
   1. **Description:** Captures events related to endpoint security solutions.
   2. **Examples:** Logs from endpoint protection platforms.

**Network Logs:**

1. **Network Device Logs:**
   1. **Description:** Captures events and errors on network devices (routers, switches, etc.).
   2. **Examples:** Router and switch logs.
2. **Firewall Logs:**
   1. **Description:** Records information about traffic allowed or denied by the firewall.
   2. **Examples:** iptables logs (Linux), Windows Firewall logs.
3. **Proxy Logs:**
   1. **Description:** Records details about web traffic, user activity, and content filtering.
   2. **Examples:** Squid proxy logs.
4. **DNS Logs:**
   1. **Description:** Captures DNS query and response information.
   2. **Examples:** Bind logs.
5. **VPN Logs:**
   1. **Description:** Logs related to Virtual Private Network (VPN) connections.
   2. **Examples:** VPN server logs.

Effective log management involves collecting, storing, and analyzing logs centrally using tools like SIEM (Security Information and Event Management) systems. Regularly reviewing logs helps identify and respond to security incidents, troubleshoot issues, and ensure compliance with security policies.

The evaluation of warning messages is a critical aspect of system monitoring and security management. Warning messages, often generated by various software applications, operating systems, and security tools, provide indications of potential issues or events that may require attention. Here are key considerations for evaluating warning messages:

**Contextual Understanding:**

**Assess the Context:** Understand the context in which the warning message was generated. Consider the application, system component, or security tool producing the warning and its role in the overall environment.

**Severity and Impact:**

**Determine Severity:** Evaluate the severity level assigned to the warning. Higher severity levels generally indicate more critical issues.

**Assess Impact:** Consider the potential impact of the warning on system performance, security, or user experience.

**Correlation with Other Events:**

**Correlate Events:** Analyze the warning message in conjunction with other log entries or events. Look for patterns or correlations that may provide a more comprehensive understanding of the situation.

**Baseline Comparison:**

**Compare to Baseline:** Compare the warning message against baseline or normal system behavior. Deviations from the baseline may indicate anomalies or potential security incidents.

**Reputation of Source:**

**Evaluate Source Reputation:** Consider the reputation and reliability of the source generating the warning message. Trusted security tools and well-established applications are generally more reliable.

**User and System Context:**

**Consider User and System Context:** Take into account the user context and the specific system where the warning occurred. Different user roles and system configurations may influence the interpretation of warnings.

**Documentation and Knowledge Base:**

**Check Documentation:** Refer to product documentation, knowledge bases, and relevant sources to understand the meaning of specific warning messages. Vendors often provide information on common warnings and their resolutions.

**False Positives:**

**Evaluate for False Positives:** Be aware of the potential for false positives. Some warnings may be triggered by benign events, misconfigurations, or temporary issues.

**Alert Thresholds:**

**Review Alert Thresholds:** Understand the thresholds set for generating warning messages. Adjust thresholds if needed to align with the specific requirements and characteristics of the environment.

**Response and Mitigation:**

**Define Response Procedures:** Establish clear procedures for responding to warning messages. Outline steps for investigation, escalation, and mitigation based on the severity and nature of the warning.

**Logging and Retention:**

**Ensure Proper Logging:** Ensure that warning messages are adequately logged, and logs are retained for an appropriate duration. This facilitates historical analysis and forensic investigations.

**Continuous Improvement:**

**Iterate and Improve:** Regularly review and refine the evaluation process based on evolving system requirements, changes in the threat landscape, and feedback from incident response activities.

By systematically evaluating warning messages using these considerations, organizations can enhance their ability to detect and respond to potential issues promptly, improving overall system security and reliability.

Working with network security data involves monitoring, analyzing, and responding to information generated by various network devices and security tools to protect against cyber threats. Here's a guide on how to effectively work with network security data:

**1. Collecting Network Security Data:**

Use network security appliances: Deploy firewalls, intrusion detection/prevention systems, and web application firewalls to generate logs and alerts.

Centralize logs: Aggregate logs from different network devices and security tools in a centralized location or SIEM (Security Information and Event Management) system.

**2. Understanding Network Topology:**

Document the network: Understand the organization's network topology, including routers, switches, firewalls, and other devices.

Identify critical assets: Determine the location of critical assets and their connectivity within the network.

**3. Monitoring Network Traffic:**

Use network monitoring tools: Employ tools like Wireshark, tcpdump, or network flow analyzers to capture and analyze network traffic.

Monitor anomalies: Set up alerts for unusual network behavior, such as unexpected traffic patterns or spikes.

**4. Analyzing Security Events:**

Correlate events: Analyze and correlate security events from different sources to identify patterns or potential security incidents.

Prioritize alerts: Prioritize alerts based on severity and impact, focusing on those that pose the most significant risk.

**5. Incident Detection and Response:**

Establish incident response procedures: Develop documented procedures for responding to network security incidents.

Investigate incidents: Thoroughly investigate security incidents to determine the scope, impact, and root causes.

**6. Utilizing Threat Intelligence:**

* Integrate threat intelligence: Incorporate threat intelligence feeds into your security infrastructure to enhance threat detection.
* Stay informed: Keep up-to-date with the latest threat intelligence to understand emerging threats and vulnerabilities.

**7. Vulnerability Management:**

* Conduct regular scans: Perform vulnerability assessments and penetration testing to identify and address potential weaknesses.
* Patch management: Keep network devices and software up-to-date with the latest security patches.

**8. Access Control and Authentication:**

* Implement strong access controls: Enforce the principle of least privilege and implement proper access controls on network devices.
* Monitor user authentication: Keep an eye on authentication logs to detect unauthorized access attempts.

**9. Logging and Auditing:**

* Enable logging: Ensure that logging is enabled on network devices, and logs are regularly reviewed.
* Audit configurations: Periodically audit network device configurations to identify and rectify security misconfigurations.

**10. Encryption and Network Segmentation:**

* Use encryption: Encrypt sensitive data in transit using protocols like HTTPS, SSL/TLS.
* Implement network segmentation: Isolate different segments of the network to limit the lateral movement of attackers.

**11. Employee Training and Awareness:**

* Educate employees: Provide training on security best practices to reduce the likelihood of social engineering attacks and insider threats.

**12. Continuous Improvement:**

* Review and adapt: Regularly review security processes, incident responses, and lessons learned to improve overall network security posture.

**13. Compliance and Reporting:**

* Ensure compliance: Understand and adhere to relevant compliance standards and regulations.
* Generate reports: Produce regular reports on network security metrics and incidents for management and compliance purposes.
* Working with network security data requires a holistic approach, combining technology, processes, and human expertise to effectively detect, respond to, and mitigate security threats in a dynamic and evolving landscape.

**11. Employee Training and Awareness:**

* Educate employees: Provide training on security best practices to reduce the likelihood of social engineering attacks and insider threats.

**12. Continuous Improvement:**

* Review and adapt: Regularly review security processes, incident responses, and lessons learned to improve overall network security posture.

**13. Compliance and Reporting:**

* Ensure compliance: Understand and adhere to relevant compliance standards and regulations.
* Generate reports: Produce regular reports on network security metrics and incidents for management and compliance purposes.
* Working with network security data requires a holistic approach, combining technology, processes, and human expertise to effectively detect, respond to, and mitigate security threats in a dynamic and evolving landscape.
* Both ELSA and SGUIL contribute to enhancing an organization's security posture by offering robust log management and analysis capabilities. ELSA excels in log aggregation and analysis across diverse data sources, making it suitable for broader use cases beyond security. SGUIL, specifically tailored for security operations, focuses on providing a comprehensive view of security events, aiding analysts in making informed decisions during incident response.
* Choosing between ELSA and SGUIL depends on the organization's specific needs and priorities. ELSA may be preferred for its versatility and broader log management capabilities, while SGUIL might be chosen for its specialization in security-related use cases, especially when integrated with Suricata. Ultimately, the selection should align with the organization's goals, the scale of log data, and the emphasis on general log management versus security-specific requirements.

**Virus analusis**

* Malware Analysis is an important skill to have. As a quick overview, Malware Analysis is performed by the following people in the Security Industry:
* **Security Operations** teams analyze malware to write detections for malicious activity in their networks.
* **Incident Response** teams analyze malware to determine what damage has been done to an environment to remediate and revert that damage.
* **Threat Hunt** teams analyze malware to identify IOCs, which they use to hunt for malware in a network.
* **Malware Researchers** in security product vendor teams analyze malware to add detections for them in their security products.
* **Threat Research** teams in OS Vendors like Microsoft and Google analyze malware to discover the vulnerabilities exploited and add more security features to the OS/applications.

Static analisis

* Malware Analysis is an important skill to have. As a quick overview, Malware Analysis is performed by the following people in the Security Industry:
* **Security Operations** teams analyze malware to write detections for malicious activity in their networks.
* **Incident Response** teams analyze malware to determine what damage has been done to an environment to remediate and revert that damage.
* **Threat Hunt** teams analyze malware to identify IOCs, which they use to hunt for malware in a network.
* **Malware Researchers** in security product vendor teams analyze malware to add detections for them in their security products.
* **Threat Research** teams in OS Vendors like Microsoft and Google analyze malware to discover the vulnerabilities exploited and add more security features to the OS/applications.

Dynamic analisis

* Static analysis might provide us with crucial information regarding malware, but sometimes that is not enough. We might need to run the malware in a controlled environment to observe what it does in these cases. Malware can often hide its properties to thwart Static Analysis. However, in most of those cases, Dynamic Analysis can prove fruitful. Dynamic analysis techniques include running the malware in a VM, either in a manual fashion with tools installed to monitor the malware's activity or in the form of sandboxes that perform this task automatically. We will learn about some of these techniques later in this room. Once we run the malware in a controlled environment, we can use our knowledge from the Windows Forensics rooms to identify what it did in our environment. The advantage here is that since we control the environment, we can configure it to avoid noise, like activity from a legitimate user or Windows Services. Thus, everything we observe in such an environment points to malware activity, making it easier to identify what the malware did in this scenario.
* Please note that malware is like a weapon because it can produce great harm if not handled with care. For this reason, always take the following precautions while analyzing malware:
* Never analyze malware or suspected malware on a machine that does not have the sole purpose of analyzing malware.
* When not analyzing or moving malware samples around to different locations, always keep them in password-protected zip/rar or other archives so that we can avoid accidental detonation.
* Only extract the malware from this password-protected archive inside the isolated environment, and only when analyzing it.
* Create an isolated VM specifically for malware analysis, which has the capability of being reverted to a clean slate once you are done.
* Ensure that all internet connections are closed or at least monitored.
* Once you are done with malware analysis, revert the VM to its clean slate for the next malware analysis session to avoid residue from a previous malware execution corrupting the next one.
* Malware, however, often uses techniques to prevent an analyst from performing dynamic analysis. Since most dynamic analysis is performed in a controlled environment, most methods to bypass dynamic analysis include detecting the environment in which it is being run. Therefore, in these cases, the malware uses a different, benign code path if it identifies that it is being run in a controlled environment.
* In the context of cybersecurity, a **virus** is a type of malicious software (malware) that is designed to spread from one computer or device to another by attaching itself to legitimate files or programs. Once a virus infects a device, it can cause a range of harmful effects, such as deleting files, stealing personal information, or even taking control of the device.
* Viruses can spread through various means, including email attachments, downloads from the internet, or infected removable media such as USB drives. They can be spread by opening an infected file or clicking on a malicious link or attachment.
* To protect against viruses, it's important to have up-to-date antivirus software installed on your computer or device, and to exercise caution when downloading files or clicking on links from unknown sources. Regularly backing up important data can also help to mitigate the damage in case of a virus attack.

Another online service that utilises these checksums is Virustotal. Virustotal acts as an indexer and aggregator for various Anti Virus (AV) engines. When a checksum is submitted to Virsutotal, fellow malware analysts can view the AV reports attributed to that file.

Much like a search engine, you can search for reports by a few characteristics, for example:

* The IP Addresses that samples communicate with
* Checksums
* The file itself
* In the context of cybersecurity, a **worm** is a type of malicious software (malware) that is designed to spread from one device to another by exploiting vulnerabilities in a network or operating system. Unlike viruses, which require a user to open or execute an infected file, worms can spread automatically and independently, without any action on the part of the user.
* Once a worm infects a device, it can spread to other devices on the same network or connected to the internet, causing a range of harmful effects, such as slowing down network performance, stealing sensitive data, or even taking control of devices.
* Worms can exploit various vulnerabilities, such as unpatched software, weak passwords, or social engineering attacks to trick users into clicking on a malicious link or attachment.
* To protect against worms, it's important to keep software and operating systems up-to-date with the latest security patches, use strong passwords, and avoid clicking on links or opening attachments from unknown sources. Firewalls and intrusion detection systems can also help to prevent or detect the spread of worms in a network.

Virus types

**Multipartite Viruses**

* Combines features of other virus types.
* Can infect files and embed in system memory.
* Examples: Tequila, Invader.

**File Infector Viruses**

* Attaches to executable files, such as .exe and .dll.
* Spreads when infected files are executed.
* Examples: CIH (Chernobyl), Sasser.

**Macro Viruses**

* Infects documents, spreadsheets, and other macro-enabled files.
* Common in the early days of Office documents.
* Examples: Concept, Melissa.

**Resident Viruses**

* Embed themselves in a computer's memory and can infect multiple files.
* Execute when the operating system loads.

**Non-Resident Viruses**

* Do not embed in system memory but infect files directly.
* Activate when the infected file is executed.
* Examples: Vienna, Cascade.

**Virus Analysis and Detection Methods**

* Analyzing and Identifying Viruses
* Signature-Based Detection
* Behavioral Analysis
* Varieties of Viruses
* File Infector Viruses
* Macro Viruses
* Resident Viruses
* Non-Resident Viruses
* Multipartite Viruses
* In cybersecurity, a **Trojan horse** (often shortened to "Trojan") is a type of malicious software (malware) that is disguised as a legitimate program or file but actually performs a malicious action when executed.
* Trojan horses can be designed to perform a variety of harmful actions, such as stealing sensitive information, disabling security features, or allowing unauthorized access to a computer or network. They can also be used to install other types of malware, such as viruses or worms, onto a compromised device.
* Trojan horses are typically distributed through social engineering tactics, such as phishing emails or downloads from fake websites, that trick the user into installing the malware. They can also be hidden in legitimate-looking software or files, making them difficult to detect.
* To protect against Trojan horses, it's important to use antivirus software and keep it up-to-date, avoid downloading software or files from unknown sources, and be cautious of unexpected emails or messages. Regularly backing up important data can also help to mitigate the damage in case of a Trojan attack.

**Infection typically begins when a user unknowingly downloads and executes a Trojan-infected file or program.**

· Common infection vectors:

* Email attachments
* Fake software downloads
* Infected websites
* Malvertising
* Social engineering

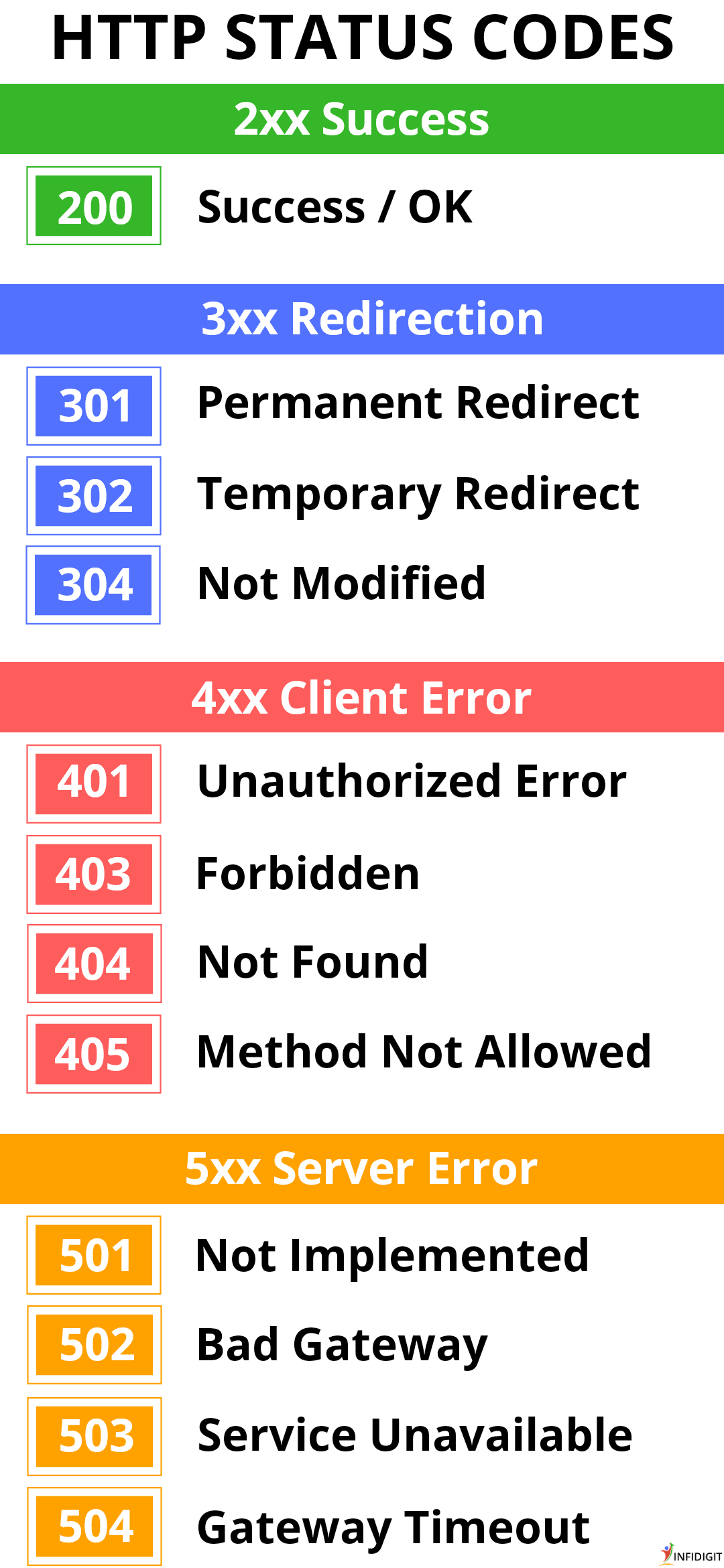
**After execution, the Trojan may perform various malicious activities silently in the background, such as:**

* Stealing data (passwords, financial information, personal data).
* Creating a backdoor for remote access.
* Downloading and installing other malware.
* Disrupting or damaging the system.

ფიშინგი ინტერნეტთაღლითობის ფორმაა, რომელიც მომხმარებელს მოტყუების გზით აიძულებს, გაამჟღავნოს თავისი სენსიტიური და პერსონალური ინფორმაცია თაღლითების მიერ შექმნილ ყალბ ვებგვერდზე შეყვანის გზით.

ფიშინგის დროს მომხმარებელთან მიდის შეტყობინება ელექტრონული ფოსტით, სოციალური ქსელით ან სხვა საშუალებით, წერილის ავტორად კი ხშირად ფინანსური ორგანიზაცია ან სხვა, მომხმარებლისათვის ნაცნობი ორგანიზაცია მიეთითება. შესაძლოა, ავტორად დაფიქსირებული იყოს ამ უკანასკნელის დასახელების იმიტაციაც. თაღლითები ცდილობენ, მოტყუებით გადაიყვანონ მომხმარებელი შეტყობინებაში მოცემულ ბმულზე, რომელიც ყალბი ვებგვერდის მისამართს წარმოადგენს, ან/და ჩამოატვირთინონ მიმაგრებული მავნე ფაილი, რომელიც წარმოდგენილია როგორც ლეგიტიმური დოკუმენტი. ამ გზით მომხმარებლის ინფორმაცია (ვებგვერდზე მომხმარებლის მიერ შეყვანილი ნებისმიერი ინფორმაცია, ასევე ჩამოტვირთული მავნე ფაილის მიერ მოპოვებული ინფორმაცია) კი თაღლითების ხელში აღმოჩნდება და საშუალებას მისცემს მათ, ეს ინფორმაცია თავისი დანაშაულებრივი მიზნებისთვის გამოიყენონ.

აღსანიშნავია, რომ თაღლითების მიერ შექმნილი ყალბი ვებგვერდი, ხშირ შემთხვევაში, ვიზუალურად მაქსიმალურადაა მიმსგავსებული ფინანსური ინსტიტუტის ნამდვილ, ოფიციალურ ვებგვერდს (მაგ. ინტერნეტბანკის ვებგვერდი, სხვადასხვა გადახდების ვებგვერდი). გაყალბებულ ვებგვერდზე, რომელიც ინტერნეტბანკის მსგავსია, მომხმარებლის სახელის (username), პაროლისა და სხვა უსაფრთხოების მონაცემების შეყვანით, თაღლითები მოიპოვებენ მომხმარებლის საბანკო ანგარიშებზე წვდომას, რაც ნიშნავს, რომ მათ შეუძლიათ მსხვერპლის სახელით სესხის აღება, მისი დეპოზიტის ხელშეკრულების დარღვევა და თანხის მითვისება, თანხების გადარიცხვა და ა.შ. ხოლო გადახდების ვებგვერდზე მომხმარებლის მიერ გამჟღავნებული ბარათის ნომრის, მოქმედების ვადის და უსაფრთხოების სამნიშნა კოდის საშუალებით თაღლითებს შესაძლოა საშუალება მიეცეთ ინტერნეტსივრცეში მომხმარებლის ბარათით განახორციელონ არასანქცირებული გადახდები.



Example of Spear Phishing

An attacker tried to target an employee of NTL World, which is a part of the Virgin Media company, using spear phishing. The attacker claimed that the victim needed to sign a new employee handbook. This was designed to lure them into clicking a link where they would have been asked to submit private information.

Vishing

Vishing, which is short for "voice phishing," is when someone uses the phone to try to steal information. The attacker may pretend to be a trusted friend or relative or to represent them.

Example of Vishing

In 2019, there was a vishing campaign that targeted members of the UK’s parliament and their staffers. The attack was part of an assault that involved at least 21 million spam emails targeting UK lawmakers.

Email Phishing

In an email phishing scam, the attacker sends an email that looks legitimate, designed to trick the recipient into entering information in reply or on a site that the hacker can use to steal or sell their data.

Example of Email Phishing

Hackers used LinkedIn to grab contact information from employees at Sony and targeted them with an email phishing campaign. They got away with over 100 terabytes of data.

HTTPS Phishing

An HTTPS phishing attack is carried out by sending the victim an email with a link to a fake website. The site may then be used to fool the victim into entering their private information.

Example of HTTPS Phishing

Hacker group Scarlet Widow searches for the employee emails of companies and then targets them with HTTPS phishing. When the user gets a mostly empty email, they click on the little link that is there, taking the first step into Scarlet Widow's web.

Pharming

In a pharming attack, the victim gets malicious code installed on their computer. This code then sends the victim to a fake website designed to gather their login credentials.

Example of Pharming

In 2007, a complex pharming attack went after at least 50 financial institutions across the world. Users were directed to false websites and instructed to enter sensitive information.

Pop-up Phishing

Pop-up phishing often uses a pop-up about a problem with your computer’s security or some other issue to trick you into clicking. You are then directed to download a file, which ends up being malware, or to call what is supposed to be a support center.

Example of Pop-up Phishing

Users have sometimes received pop-ups saying they can qualify for AppleCare renewal, which would supposedly avail them of extended protection for their Apple devices. However, the offer is fake.

Evil Twin Phishing

In an evil twin attack, the hacker sets up a false Wi-Fi network that looks real. If someone logs in to it and enters sensitive details, the hacker captures their info.

Example of Evil Twin Phishing

A Russian military agency called GRU was recently charged with executing evil twin attacks using fake access points. The access points were made to look like they provided connections to real networks when in reality they led users to sites that stole their credentials or downloaded malware onto their computers.

Watering Hole Phishing

In a watering hole phishing attack, a hacker figures out a site a group of users tends to visit. They then use it to infect the users’ computers in an attempt to penetrate the network.

Example of Watering Hole Phishing

In 2012, the U.S. Council on Foreign Relations was targeted by a watering hole attack. The assault aimed to take advantage of the high-profile users that were frequenting the site, as well as the login credentials they could provide. The attack achieved some success, particularly using a vulnerability within Internet Explorer.

Whaling

A whaling attack is a phishing attack that targets a senior executive. These individuals often have deep access to sensitive areas of the network, so a successful attack can result in access to valuable info.

Example of Whaling

A founder of Levitas, an Australian hedge fund was the target of a whaling attack that led the individual to a fake connection using a fraudulent Zoom link. After following the link, they had malware installed on their system, and the company lost $800.000.

Clone Phishing

A clone phishing attack involves a hacker making an identical copy of a message the recipient already received. They may include something like “resending this” and put a malicious link in the email.

Example of Clone Phishing

In a recent attack, a hacker copied the information from a previous email and used the same name as a legitimate contact that had messaged the victim about a deal. The hacker pretended to be a CEO named Giles Garcia and referenced the email Mr. Garcia had previously sent. The hacker then proceeded to pretend to carry on the previous conversation with the target, as if they really were Giles Garcia.

Deceptive Phishing

Deceptive phishers use deceptive technology to pretend they are with a real company to inform the targets they are already experiencing a cyberattack. The users then click on a malicious link, infecting their computer.

Example of Deceptive Phishing

Users were sent emails that came from the address support@apple.com and had “Apple Support” in the sender information. The message claimed that the victim’s Apple ID had been blocked. They were then prompted to validate their accounts by entering information the hacker would use to crack it.

Social Engineering

Social engineering attacks pressure someone into revealing sensitive information by manipulating them psychologically.

Example of Social Engineering

A hacker pretended to be a representative of Chase Bank while saying that the action was needed on the target’s debit or ATM card. The attacker was trying to pressure the victim into divulging their information by leveraging their fear of not being able to access their money in their Chase account.

Angler Phishing

Anglers use fake social media posts to get people to provide login info or download malware.

Example of Angler Phishing

Hackers pretended to represent Domino's Pizza on Twitter, fielding the concerns and comments of customers. Once they engaged with a customer, they would use their situation to try to get their personal information—using the guise of trying to get them a refund or a reward.

Smishing

Smishing is phishing through some form of a text message or SMS.

Example of Smishing

Hackers pretended to be from American Express and sent text messages to their victims telling them they needed to tend to their accounts. The message said it was urgent, and if the victim clicked, they would be taken to a fake site where they would enter their personal information.

Man-in-the-Middle (MTM) Attacks

With a man-in-the-middle attack, the hacker gets in “the middle” of two parties and tries to steal information exchanged between them, such as account credentials.

Example of Man-in-the-Middle-Attack

In 2017, Equifax, the popular credit score company, was targeted by man-in-the-middle attacks that victimized users who used the Equifax app without using HTTPS, which is a secure way to browse the internet. As the users accessed their accounts, the hackers intercepted their transmissions, stealing their login credentials.

Website Spoofing

With website spoofing, a hacker creates a fake website that looks legitimate. When you use the site to log in to an account, your info is collected by the attacker.

Example of Website Spoofing

Hackers made a fake Amazon website that looked nearly identical to the real Amazon.com but had a different Uniform Resource Locator (URL). All other details, including fonts and images, looked legitimate. Attackers were hoping that users would put in their username and password.

Domain Spoofing

Domain spoofing, also referred to as DNS spoofing, is when a hacker imitates the domain of a company—either using email or a fake website—to lure people into entering sensitive information. To prevent domain spoofing, you should double-check the source of every link and email.

Example of Domain Spoofing

An attacker would execute a domain spoofing attack by creating a fraudulent domain made to look like a real LinkedIn site, for example. When users go to the site and enter any information, it is sent straight to hackers who could use it or sell it to someone else.

Image Phishing

Image phishing uses images with malicious files in them meant to help a hacker steal your account info or infect your computer.

Example of Image Phishing

Hackers have made use of AdGholas to hide malicious code written in JavaScript inside images and HTML files. When someone clicked on an image generated by AdGholas, malware would be downloaded onto their computer that could be used to phish for their personal information.

Search Engine Phishing

A search engine phishing attack involves an attacker making fake products that look attractive. When these pop up in a search engine, the target is asked to enter sensitive information before purchasing, which then goes to a hacker.

Example of Search Engine Phishing

In 2020, Google said that they found 25 billion spam pages every day, like the one put up by hackers pretending to be from the travel company Booking.com. An ad would pop up in users’ search results that looked like it was from booking.com and included the site’s address and the kind of wording users would expect from a real ad by the company. After users clicked, they were prompted to enter sensitive login information that was then transmitted to hackers.