

## Sieć Tor, anonimizacja ruchu

### Rys historyczny

Rozwój sieci Internet w latach 90-tych spowodował konieczność wprowadzenia zabezpieczeń transmisji danych. Wkrótce okazało się, że oprócz danych (które można zaszyfrować) metadane, czyli informacje kto, kiedy, skąd, dokąd i jaką objętość danych przesłał są również możliwe do przechwycenia i wykorzystania. Stąd potrzeba zachowania prywatności i w tym zakresie.

Podstawy teoretyczne sieci Tor powstały już w początku lat 90-tych. W 1994 zaproponowano użycie wielokrotnego, ustandaryzowanego szyfrowania jako mechanizmu zabezpieczenia komunikatu na drodze wieloetapowego routingu [van Oorschot P. - "On unifying some cryptographic protocol logics"], a w roku 1998 powstało gotowe teoretyczne opracowanie trasowania cebulowego [Reed M. G., Syverson P.F., Goldschlag D.M. - „Anonymous connections and onion routing”]. Był to wynik projektu wojskowego (amerykański ośrodek Office of Naval Research, następnie DARPA) mającego na celu zabezpieczenie komunikacji w otwartych sieciach między systemami militarnymi i wywiadowczymi jak i zapewnienie anonimizacji ruchu operacyjnego.

Praktyka jednak pokazała, że działanie tego rodzaju sieci wymaga maksymalnego rozproszenia (i geograficznego i pod względem routingu) jej węzłów, co nie jest wykonalne dla jednej organizacji, nawet armii USA, stąd udostępnienie rozwiązań na potrzeby Internetu. Pierwsza wersja testowej implementacji Tor pojawiła się więc w 2002 roku, z wydaniem wersji 1.0 w 2003.

W 2004 wydano wersję o otwartych źródłach. Dzięki temu możliwe było przeprowadzenie odpowiedniego audytu oprogramowania, którego pochodzenie dyskwalifikowało je z pełnego zaufania, i przede wszystkim wprowadzenie odpowiednich poprawek. Wkrótce pojawiły się pakiety oprogramowania umożliwiające anonimowe korzystanie z sieci bez zaawansowanych umiejętności - najbardziej popularnym wówczas był pakiet łączący przeglądarkę Opera z programem Tor (OperaTOR), jednak ze względu na zamknięte źródła przeglądarki i rosnące użycie technologii JavaScript nie oferował on satysfakcjonującego poziomu bezpieczeństwa.

Powstałe od 2006 roku wydania bazujące na przeglądarce Firefox umożliwiają blokowanie tego rodzaju technik co znacząco zwiększa bezpieczeństwo - dzięki blokadzie JavaScript powierzchnia ataku na klienta przez użycie złośliwego skryptu została znacząco zmniejszona. Od tej pory popularność sieci Tor ciągle rośnie, pośrednio wskutek rosnącej świadomości użytkowników Internetu, a w dużej mierze w wyniku zapotrzebowania na ochronę przed inwigilacją - tą od dostawców usług oraz od organów opresji, jak i koniecznością ominięcia cenzury Internetu występującej już w większości krajów świata. Zdolność kredytową, wysokość ubezpieczeń czy punktację kredytu społecznego już liczy się na podstawie historii szukanych fraz i odwiedzanych stron, stąd coraz popularniejsze jest zachowywanie prywatności za pomocą technologii Tor. Anonimizacja chroni również przed „informacyjną bańką” stosowaną w tzw. sieciach społecznościowych.

### Czym jest sieć TOR?

TOR (ang. The Onion Routing - tłumaczone jako "Trasowanie cebulowe") jest usługą działającą w sieci Internet zapewniającą wyższą anonimowość niż w przypadku zwykłej sieci. Zastosowanie TOR ukrywa adres IP klienta i serwera, a dodatkowo utrudnia namierzenie tych elementów przez analizę ruchu sieciowego. Szyfrowanie ruchu uniemożliwia podsłuchiwanie przez ISP. Przekazywanie pakietów następuje ustaloną przed wysłaniem trasą za pomocą łańcucha węzłów.

Ze względu na to, że w przypadku ukrytej usługi Tor ukrywa również IP serwera (jak i IP użytkownika dla serwera nie jest widoczne), użycie tej technologii pozwala ominąć cenzurę niektórych usług w sieci. Z drugiej strony jednak wiele usług, których działanie opiera się na pozyskiwaniu i przetwarzaniu danych użytkowników stara się blokować anonimizowany ruch z sieci Tor.

Tor i usługi przezeń działające używane są więc m. in. do anonimowej komunikacji, obchodzenia cenzury, zgłaszania wycieków, ukrywania zawartości pakietów przed głęboką analizą czy ochrony przed inwigilacją. Wiele serwisów w „otwartej” sieci ma swoje odpowiedniki w sieci Tor, jak i istnieją specjalne serwisy wyłączne w sieci Tor.

Przy korzystaniu z sieci Tor należy wciąż pamiętać o OPSEC, czyli ogólnych zasadach bezpieczeństwa pracy, by samemu nie upubliczniać informacji mogących nas zdemaskować - warto tutaj zrealizować sobie na potrzeby TOR oddzielną tożsamość. Każda informacja wprowadzona równoległe do sieci Tor i sieci otwartej może zdemaskować wprowadzającego. Podobnie działają metadane: Sposób pisania, numer seryjny aparatu fotograficznego zapisany w plikach zdjęć, układ uszkodzonych pikseli na matrycy, daty i godziny plików, wersja programu tworzącego archiwum to wszystko informacje umożliwiające powiązanie tożsamości z sieci tor i sieci otwartej. Firefox rozpowszechniany w pakiecie z Tor posiada pewne standardowe wymiary okienka, których **nie należy zmieniać**, bowiem skrypty mogą je odczytać. Nie należy również logować się do kont, na które logowaliśmy się czy które zakładaliśmy z sieci otwartej. Jeżeli już musimy eksponować przez Tor takie usługi jak np. SSH, minimum bezpieczeństwa to uwierzytelnianie kluczem (a nie hasłem) oraz pakiet fail2ban, który zmusi atakującego do zmieniania trasy co kilka prób.

## Typy węzłów

W strukturze sieci wyróżniamy trzy rodzaje komputerów-przebieźców:

- **Guard relay (Entry node)** - punkt wejścia do sieci, z którymi łączy się komputer kliencki.
- **Middle relay** - punkty, między którymi wymieniane są zaszyfrowane pakiety. Węzły te nie mają informacji o zawartości pakietów - każdy ma tylko informację gdzie przekazać pakiet.
- **Exit node** - Węzły wyjściowe - odszyfrowują ruch i kierują go do docelowego miejsca, np. usługi w "jawnej" sieci WWW. Są one uruchamiane zazwyczaj w miejscach o wysokiej wolności cyfrowej i niskim poziomie rządowej inwigilacji bądź na preferencyjnych warunkach. IP węzła wyjściowego to IP widziane przez docelowy serwer, najczęściej pod sporym nadzorem. Węzeł taki „bierze na siebie” cały ruch wynikowy. Dopóki więc nie jesteśmy w jakimś bezpiecznym pod względem Internetu kraju lub nie przejęliśmy cudzego serwera, nie uruchamiamy węzła wyjściowego na swoim serwerze.

## Działanie sieci

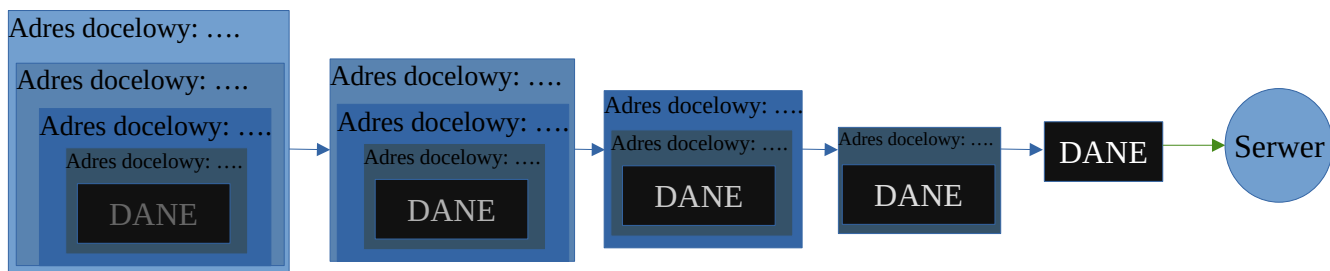
Ruch sieciowy w sieci Tor jest szyfrowany, a następnie kierowany losową, wyznaczoną przed nawiązaniem połączenia trasą.

Klient chcąc wysłać pakiet najpierw szyfruje go w taki sposób, by tylko węzeł wyjściowy mógł go odszyfrować. Następnie dołącza jego adres i szyfruje pakiet ponownie, kluczem węzła poprzedniego w trasie pakietu, po czym znów dołącza adres węzła i szyfruje kluczem poprzedniego itd. - szyfrowanie to powtarza się do osiągnięcia węzła wejściowego. Następnie tak wielokrotnie szyfrowany pakiet jest wysyłany do węzła wejściowego (Entry node).

Węzeł wejściowy odszyfrowuje swoją warstwę szyfrowania uzyskując adres docelowego węzła pośredniego oraz dane zaszyfrowane. Te dane są przesyłane do kolejnego, wskazanego w deszyfrowanym pakiecie węzła.

Kolejny węzeł po uzyskaniu pakietu znów odszyfrowuje i przesyła pakiet dalej.

Sytuacja powtarza się do osiągnięcia exit node'a, który wysyła "surowe" dane do serwera docelowego „otwartym” Internetem (na rysunku 1 - zielona strzałka). W dziennikach dostępu serwera widoczny jest więc adres IP punktu wyjściowego (Exit Node). ISP użytkownika zaś widzi wyłącznie zaszyfrowany ruch do jakiegoś punktu. Taki ruch może zostać adaptowany dodając do niego nieużywane nagłówki, co w przypadku głębokiej inspekcji pakietów spowoduje, że zostanie wzięty np. za komunikację gry online, HTTPS, standardowego połączenia P2P, czy rozmowy przez komunikator.



Rys. 1 - Ramka sieci Tor deszyfrowana w kolejnych stadiach transmisji. Każdy prostokąt to warstwa szyfrowania.

W ten sposób każdy węzeł sieci otrzymuje tylko niezbędne do działania informacje - zasada "**need to know**" - i w przeciwieństwie do popularnego przy przetwarzaniu danych w usługach sieci Internet bezpieczeństwa opartego na próżnych zapewnieniach (ich słabą skuteczność pokazało zajęcie siedzib usług sieciowych działających w Afganistanie w roku 2021 - firmy deklarujące w regulaminach „nie przetwarzanie” danych okazały się je posiadać i udostępniać), tutaj nie jest możliwe poznanie całości trasy bez pełnego ataku na kilka warstw szyfrowania (lub przejścia wszystkich węzłów). Szyfrowanie zapewniane jest do osiągnięcia exit node'a. Jeżeli dane na poziomie łączności exit node'a z usługą są jakoś zaszyfrowane (np. HTTPS), exit node również nie dysponuje informacjami wymienianymi z komputerem docelowym choć może próbować ataków umożliwiających pozyskanie takich informacji. Stąd **nie należy łączyć się przez Tor do serwerów typu "Telnet" czy "FTP"**, jak i sprawdzać poczty na serwerze bez SSL, bo przesyłane hasło zostanie przechwycone przez przejęty exit node. Również, jakkolwiek w Linuxie możliwe jest (polecenie **torify**) „zawinięcie” w Tor każdego programu, bardzo **niepolecane jest stosowanie trasowania Tor do anonimizacji istniejących implementacji sieci P2P**, bo i tak na końcu, po odszyfrowaniu pakietu prawdziwy adres IP będzie w nim uwidoczny (co wynika z protokołu np. sieci Torrent czy eD2k). Ze względu na możliwość ataków korelacyjnych związanych z modelowaniem czasów odszyfrowania przez różne trasy, nieraz przed wysłaniem odszyfrowanego pakietu węzeł czeka losowe opóźnienie. Jest to całkowicie intencjonalne działanie. Trasa pakietów również jest periodycznie zmieniana.

Przy uruchomieniu, Tor musi uzyskać informację na temat działających punktów wejścia do sieci by się do nich połączyć. Istnieją dwa rodzaje tego rodzaju punktów:

- **Punkty ogólnodostępne** - pozyskiwane przez Tor co uruchomienie za pomocą istniejącej listy wgranej w program. Program łączy się z usługą katalogową i pozyskuje adresy.
- Mostki - **Bridges** - punkty niejawne w celu skorzystania awaryjnego, gdy np. punkty wejściowe są blokowane na państwowym firewallu. Adresy tych punktów są pozyskiwane z bazy w niewielkich ilościach i mogą być nawet wprowadzane manualnie, co umożliwia przekazanie ich np. na papierze.

Kolejnym problemem jest konieczność zestawienia trasy. Inwentarz działających węzłów przechowywany jest na 10 podstawowych węzłach katalogowych (z czego jeden synchronizuje bazę bridge'ów) - **Directory Authorities** - położonych w różnych miejscach na świecie. Lista ta jest okresowo aktualizowana i synchronizowana przez wszystkie DA.

Synchronizacja opiera się na zasadzie **konsensusu** - by node dostał się na listę, musi być osiągalny przez większość DA. Dzięki temu nie są uwzględniane węzły z ograniczeniami geograficznymi, działające w celu krótkotrwałych kampanii demaskowania użytkowników np. na terenie jednego kraju, czy skonfigurowane na ruch z jednego punktu.

## Usługi jawne i ukryte

Łączenie z użyciem exit node'ów do "otwartej" sieci Internet to połączenie do usługi jawnej. Usługi dostępne wyłącznie w sieci Tor to tzw. usługi ukryte (**hidden services**) lub usługi Onion (**Onion services**). Usługi te mają charakterystyczne, długie (56 znaków) adresy z końcówką **.onion**. Wprowadzenie tych adresów w pole adresu przeglądarki Tor spowoduje połączenie się do tej usługi. Wymaganiem protokołu jest, by adresy kończył znak wersji (tutaj „d”, czyli licząc od zera - „a” wersja 3). Adresy zawierają niezbędne informacje kontrolne dla offline-owego sprawdzania ich poprawności.

**Uwaga:** Adresy 16-znakowe (v. 2), ze względu na możliwości kolizji (czyli 2 różnych kluczy skracanych do tego samego adresu), zostały wycofane w 2021 roku.

Na przykład adres onion strony projektu Tor to:

**`http://2gzyxa5ihm7nsggfnu52rck2vv4rvmdlkiu3zzui5du4xyc1en53wid.onion/`**

Taki adres .onion jest całością publicznego klucza, jakim zaszyfrować należy dane, by serwer usługi mógł je odszyfrować. Starsze (v. 2) adresy .onion były skrótami klucza i musiały być sprawdzone w usłudze katalogowej. Współcześnie usługa katalogowa adresów służy jedynie do uzgodnienia trasy.

Możliwe jest metodą „prób i błędów” wygenerowanie adresu zawierającego kilka znaków w ustalonych miejscach - w przypadku adresów v. 2 czyniło to je łatwiejszymi do zapamiętania, w przypadku adresów v. 3 niemożność szybkiego zweryfikowania „z pamięci” całości adresu powoduje, że teoretycznie możliwe jest podszycie się pod usługę, a użytkownik będzie weryfikował jedynie znajome mu znaki, stąd nie jest polecane stosowanie takich adresów v.3.

## Jak serwer staje się usługą?

Serwer nawiązuje połączenie z siecią Tor. Część połączeń z węzłami zostaje zachowana, węzły te spełniają wówczas rolę punktów wprowadzenia usługi (**introduction points**). Domyślnie jest to połączenie do trzech węzłów, również zabezpieczone tak jak ruch kliencki.

Następnie lista punktów wprowadzenia jest podpisywana cyfrowo kluczem usługi i umieszczana w rozproszonej tablicy (DHT). By serwer nie ujawnił swojego położenia umieszczenie to jest również anonimizowane jak ruch kliencki. Węzły pośrednie na linii „Serwer - Punkt wprowadzenia” są znane jedynie temu węzłowi i serwerowi, a klient kontaktuje się z introduction point lub rendezvous point.

## Klient dostaje się do serwera

Gdy zatwierdzony zostaje adres usługi, klient uzyskuje z DHT odpowiednią listę węzłów wprowadzających. Korzystając z klucza usługi może zweryfikować autentyczność tej listy.

Przed zestawieniem łączy do węzła wprowadzającego klient rozpoczyna **procedurę Rendezvous**.

Przekazuje jednorazowy kod wybranemu węzłowi pośredniemu sieci i nawiązuje połączenie do węzła wprowadzającego serwer ustawiając do niego trasę.

Po otrzymaniu wiadomości zaszyfrowanej swoim kluczem, serwer odszyfrowuje ją i uzyskuje kod jednorazowy, który **weryfikuje z węzłem pośrednim**, który uzyskał ten kod od klienta. Dzięki temu:

- Istnieje dwustronna pewność - i klienta i serwera poparta alternatywną drogą przekazania sekretu.
- Można rozproszyc ruch na dodatkowy węzeł, dzięki czemu nie obciążamy węzłów wprowadzających, których mamy tylko trzy.

Proszę zauważyć, że połączenia zaczynają się „od serwera” - właśnie przy okazji "**przebiliśmy**" się **przez NAT**. Korzystając z usługi Onion można więc ustawić sobie serwer pomimo tego, że jesteśmy za NAT-em.

Duża anonimowość w tego rodzaju sieciach powoduje jednak, że chcąc postawić taki serwer na własne potrzeby (np. w domu) musimy zadbać o parę spraw:

- **Limitowanie łącza** (np. programem WonderShaper) w celu uniknięcia „zatkania” łącza przy dużym natężeniu ruchu, również w celu uniknięcia podejrzeń ze strony ISP.
- **Bezpieczeństwo serwera i jego programów** - Ze względu na specyfikę sieci, kwestia ataku na taką usługę czy wsadowego testowania podatności to nie „czy”, ale „kiedy” atak nastąpi.

## Możliwe ataki

Jakkolwiek masowa deanonimizacja wszystkich użytkowników sieci jest trudna, istnieje możliwość pewnych ataków, przy czym są one zazwyczaj dość kosztowne:

- **Atak na klienta** - często stosowana metoda, komputer kliencki może zostać skompromitowany przez uruchomienie na nim nieautoryzowanego kodu korzystając z innej luki zabezpieczeń.
- **Atak na przeglądarkę klienta** - zmuszenie przeglądarki wykorzystywanej przez Tor do udzielenia informacji o komputerze, jego prawidłowego adresu IP, wysłania zapytania do DNS (zakładając, że mamy kontrolę nad serwerem DNS można wówczas uzyskać precyzyjną deanonimizację), podania informacji dotyczącej systemu klienckiego, "browser fingerprinting".
- **Atak na szyfrowanie** - współcześnie nie wykorzystywane na szeroką skalę, jednak można magazynować ruch "na później", gdy będą dostępne wystarczająco szybkie komputery lub pojawiają się podatności w algorytmach.
- **Podstawienie węzła wyjściowego** - węzeł taki może próbować np. downgrade'ować zabezpieczenia konwencjonalne (SSL, TLS) do poziomu umożliwiającego łatwiejsze złamanie.
- **Zalewanie węzłami** - dążenie, by połączenie do usługi występowało wyłącznie przez węzły będące pod kontrolą atakującego. Możliwe do wykonania "zalewając" sieć tysiącami serwerów o preferowanych parametrach (np. szybkim łączu).
- **Atak korelacyjny** - Wysyłanie dużej ilości danych "z jednej strony" sieci skutkuje przyjściem dużej ilości danych "z drugiej strony" (volume fingerprinting). Zapisując całkowity ruch wchodzący i wychodzący do sieci możliwe jest określenie jej zachowania i ustalenie zbliżonych punktów wejścia i wyjścia (circuit fingerprinting). Do niedawna stosowano również charakterystyki czasowe węzłów.
- **Korelacja przez użycie pakietów "relay early"** - Po podszyciu się za przekaźnik serwera katalogowego, wysyłany jest strumień pakietów z flagami „relay early” (część protokołu Tor-a) ustawionymi w charakterystyczny sposób (zawartość pakietu, czyli odpowiedź, musi być prawidłowa, inaczej pakiet zostanie upuszczony). Jeżeli węzeł wyjściowy atakowanych użytkowników jest pod kontrolą atakującego, można na nim sprawdzać gdzie pojawią się "znaczone" zestawy pakietów. Wymaga to długotrwałego istnienia tych węzłów - na tyle długiego by uznane zostały za przekaźniki informacji z DA i za węzły wyjściowe.

Ataki utrudniające pracę sieci i z niej korzystanie:

- **Atak na konsensus** - blokowanie działań DA. Należy podszyc się pod serwer DA lub przekaźnik z tego serwera i przesyłać do innych DA fałszywe komunikaty. Ten atak wymaga znacznej ilości przejętych komputerów (kilkanaście-kilkadziesiąt % sieci).
- **DoS** - elementy takie jak serwery DA, kluczowe węzły katalogowe adresów .onion, najważniejsze punkty wejściowe i wyjściowe mogą być obiektem ataków DoS. W styczniu 2021 wystarczyło masowe wysyłanie zapytania na serwery katalogowe adresów przez kilka

tysiący komputerów - spowodowało to brak możliwości wejścia na dowolny adres .onion korzystając z tego serwera usługi katalogowej, stan ten trwał przez kilka godzin.

- **Phishing** - Stworzenie dużej ilości podobnie wyglądających adresów i zalanie nimi otwartej sieci udając inną znaną usługę z darknetu.

### **Implementacje o jeszcze wyższym stopniu zabezpieczeń**

Problemem, do rozwiązania którego Tor nie został stworzony, jest centralizacja serwera - wciąż jest to jeden punkt, którego naruszenie powoduje zamknięcie usługi. Rozwiązania o zwiększonej nadmiarowości serwerów stosuje się w przypadkach, w których:

- Istnieje kampania cenzurująca zawartość serwerów na całym świecie.
- Strona atakująca ma środki na przeprowadzenie któregoś z ataków korelacyjnych i jest wysoce prawdopodobne, że je przeprowadzi.
- Ukrycie informacji o położeniu serwera jest "mission-critical", a strona atakująca może w celu jego namierzenia np. przejąć serwery w całych krajach (w niektórych przypadkach ujawnienie serwera może skończyć się np. ujawnieniem położenia agenta czy nawet nalotem bombowym).

O ile jednak standardową usługę można założyć na dowolnym komputerze (a nawet minikomputerze typu Raspberry Pi), o tyle wzmocnione wersje zazwyczaj wymagają znacznie większych środków.

Implementacja jednej z takich wersji oparta jest o sieć przejętych bądź udostępnionych wolontaryjnie komputerów (botnet). Komputery te:

- Stanowią pomost pomiędzy siecią Tor a siecią bezserwerową typu IPFS, w której udostępniane zasoby ciągle są wymieniane.
- Same nieustannie wymieniają ruch - stąd na żadnym z komputerów nie występuje całość udostępnianego materiału, a wszystko jest zaszyfrowane.

Większa ilość komputerów w botnecie i ich geograficzne rozproszenie gwarantuje nadmiarowość danych w przypadku utraty komputerów. Przez Tor przesyłane są jedynie zaszyfrowane dane.

Usługi tego typu **nie są zgodne** z typowym pakietem Tor i często wymagają dodatkowej wtyczki do przeglądarki, która deszyfruje ruch otrzymany z sieci rozproszonej.

Sposób ten stosowany był głównie w komunikacji wojskowej (USA, Ukraina), akcjach wywiadowczych (USA, Rosja) oraz w krajach, gdzie narodowe firewalle utrudniają komunikację na zewnątrz lub do wewnątrz (Chiny, kraje arabskie). Należy mieć na uwadze, że słaby audyt tych wtyczek był nieraz przyczyną **ataków na komputery klienckie** lub serwerowe.

Przykładowe nawiązanie połączenia w rozszerzeniu używanym przez opozycję w krajach arabskich („Halummu”, 2018):

1. Dodatek wylicza jedną z przejętych domen korzystając z algorytmu - ze względu na cenzurę nie umieszcza się domen wprost w kodzie dodatku.
2. Następuje połączenie do tej domeny przez sieć Tor.
3. Na przejętym serwerze, w podkatalogu działa skrypt komunikujący się z innymi przejętymi domenami cyrkulując zaszyfrowane zasoby. Pozyskiwana jest niezbędna strona.
4. Przejęty serwer buforuje żądany materiał i przesyła go przez sieć Tor.
5. Komputer kliencki odbiera materiał i go deszyfruje.

## LABORATORIUM:

Niezbędne elementy:

- Komputer windowsowy z maszyną wirtualną-serwerem (dalej: Serwer) - dowolny na laboratorium.
- Komputer windowsowy jako klient - dowolny na laboratorium.
- Sieć laboratoryjna.
- Opcjonalnie - możliwość dostępu do sieci Tor z innego połączenia sieciowego, z innego urządzenia (np. laptopa podłączonego do sieci Wi-fi, telefonu z własnym dostępem do Internetu i odpowiednim oprogramowaniem, modemu typu Aero2 itp.).

**UWAGA:** W przypadku telefonów komórkowych z zamkniętymi źródłowymi systemami (iOS, większość dystrybucji Android), do połączenia przez Tor należy wykorzystać opcję „Tethering”, a następnie połączyć się do Tor z komputera połączonego tak do tego telefonu. Motywowane jest to warunkami usług (Terms of Service) w tych systemach niezgodnymi z podstawowymi prawami.

Od Prowadzącego: Port usługi: .....  
Rozmiar pliku testowego dla serwera:.....M, parametr count: .....

### Część pierwsza: Klient - korzystanie z sieci Tor

Na komputerze klienckim pobieramy i rozpakowujemy (nie tworzymy skrótów menu Start!) do katalogu na dysku D/E: pakiet Tor Browser (<https://www.torproject.org/>). Jest to dystrybucja Firefoksa z usługami Tor-a. Uruchamiamy program, klikamy „Połącz” i czekamy na połączenie.

1. Sprawdzić połączenie próbując wejść na dowolną otwartą usługę (dowolną stronę internetową w otwartym Internecie). Przy użyciu dowolnie wybranego narzędzia webowego sprawdzić i zanotować swoje IP widziane przez serwery, do których łączy się Tor browser. O ile jeszcze istnieje, użyteczne jest narzędzie spod adresu <https://whatismyipaddress.com/> ze względu na możliwość zanotowania położenia węzła wyjściowego. Klikając w ikonę z lewej strony na pasku adresu („kłódka” lub favicon) sprawdzić i zanotować trasę pakietu, zrobić zrzuty ekranu do sprawozdania.

### Pomiary z następnych punktów powtórzyć 3 razy dla każdej trasy.

2. Pobrać plik mający kilkadziesiąt MB (najlepiej mały obraz jakiegoś Linuksa: <https://deb.debian.org/debian/dists/buster/main/installer-i386/> ). Zanotować adres pliku, jego rozmiar i zmierzyć czas pobierania pliku (Uwaga: Firefox pobiera od momentu **wyświetlenia okna wyboru** otwarcia lub zapisu pliku - okienka z przyciskami „radio”!). Skopiować adres pliku do schowka! **Czas pobierania pliku** można zmierzyć aplikacją „Alarmy i zegar” w systemie Windows lub za pomocą osobnego stopera np. na drugim komputerze lub we własnym PDA. Dla oszczędzenia miejsca na dysku pliki należy w kolejnych pobraniach **nadpisywać**.

3. Zmienić trasę połączenia: Kliknąć Menu z prawej strony ->New Tor circuit for this site. Sprawdzić i zanotować ponownie numer IP. Pobrać ten sam plik dokonując analogicznych pomiarów.

4. Kliknąć Menu->New identity (lub ikonka "mioteczki" na pasku narzędzi). Tor zostanie ponownie uruchomiony. Należy tutaj zauważyć stan historii używanych stron. Domyślnie Tor browser nie zapisuje historii.

**Do opracowania w sprawozdaniu:** Czasy pobierania plików vs numery IP i ich lokalizacje.

## Część druga: Serwer - Uruchamiamy ukrytą usługę

(↵ - Enter na konsoli)

0. Ze strony <http://heavy.metal.agh.edu.pl/> pobrać obraz maszyny wirtualnej serwera na **dysk D/E**:. Uruchomić VirtualBox i wgrać plik .ova zawierający maszynę wirtualną serwera (menu File → Import Virtual Appliance / Plik → Importuj Urządzenie Wirtualne). Zastartować maszynę z ustawieniami domyślnymi i zalogować się na roota.

**Login: root      Hasło: 1234**

Jeżeli mielibyśmy do czynienia z oddzielnym serwerem, można administrować nim po SSH, ale tym razem zrobimy to przez interakcję z maszyną wirtualną. Pamiętajmy, że **mając konto root musimy uważać na to co robimy, bowiem łatwo o uszkodzenie**.

1. Zainstalować Tor-a wydając polecenie:

```
apt-get install tor ↵
```

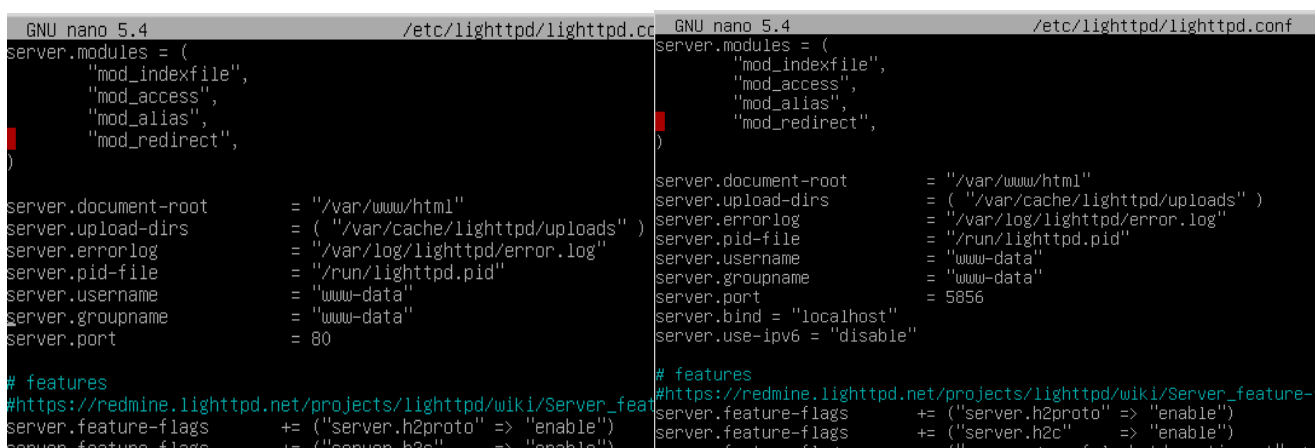
2. Skonfigurować serwer lighttpd: Edytujemy (np. edytorem nano) plik `/etc/lighttpd/lighttpd.conf`

Znaleźć blok zawierający linie `server.[...] = ....`

Dopisać do niego (lub podmieniamy wartości w odpowiednich liniach):

```
server.bind = "localhost"
server.port = 5856
server.use-ipv6 = "disable"
```

**UWAGA: port (tutaj 5856) ustalany jest przez Prowadzącego na zajęciach.**



```
GNU nano 5.4 /etc/lighttpd/lighttpd.conf
server.modules = (
    "mod_indexfile",
    "mod_access",
    "mod_alias",
    "mod_redirect",
)
server.document-root = "/var/www/html"
server.upload-dirs = ( "/var/cache/lighttpd/uploads" )
server.errorlog = "/var/log/lighttpd/error.log"
server.pid-file = "/run/lighttpd.pid"
server.username = "www-data"
server.groupname = "www-data"
server.port = 80
# features
#https://redmine.lighttpd.net/projects/lighttpd/wiki/Server_features
server.feature-flags += ("server.h2proto" => "enable")
server.feature-flags += ("server.h2c" => "enable")

GNU nano 5.4 /etc/lighttpd/lighttpd.conf
server.modules = (
    "mod_indexfile",
    "mod_access",
    "mod_alias",
    "mod_redirect",
)
server.document-root = "/var/www/html"
server.upload-dirs = ( "/var/cache/lighttpd/uploads" )
server.errorlog = "/var/log/lighttpd/error.log"
server.pid-file = "/run/lighttpd.pid"
server.username = "www-data"
server.groupname = "www-data"
server.port = 5856
server.bind = "localhost"
server.use-ipv6 = "disable"
# features
#https://redmine.lighttpd.net/projects/lighttpd/wiki/Server_features
server.feature-flags += ("server.h2proto" => "enable")
server.feature-flags += ("server.h2c" => "enable")
server.feature-flags += ("server.async-file-pushback" => "enable")
```

### Przed/po

Po zapisaniu pliku i opuszczeniu edytora przeładowujemy serwer by zaciągnął nową konfigurację poleceniami (pierwsze włącza serwer, ale z reguły jest on już włączony):

```
systemctl enable lighttpd ↵
```

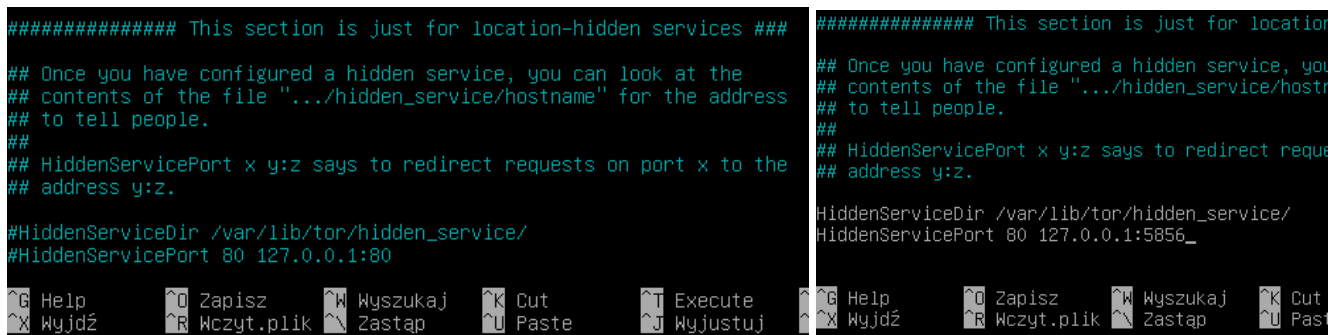
```
systemctl restart lighttpd ↵
```



3. Sprzęgnąć Tor z serwerem. Ruch dostający się na Tor na porcie **80** musi być obsługiwany przez serwer na wskazanym przez Prowadzącego porcie. Tym razem edytujemy plik **/etc/tor/torrc** . Linie występują **nżej** w pliku (Ctrl-W - wyszukiwanie w nano) i są zakomentowane znakiem „#” - należy je odkomentować kasując go.

```
HiddenServiceDir /var/lib/tor/hidden_service
HiddenServicePort 80 localhost:5856
```

Należy znaleźć pierwszą linijkę i zanotować katalog konfiguracyjny (tutaj /var/lib/tor/hidden\_service). Kolejną linijkę należy sprawdzić i doprowadzić do postaci przedstawionej w powyższym przykładzie (oczywiście z odpowiednim numerem portu, który wpisany został w punkcie 2).



The image shows two side-by-side screenshots of a nano editor window. The left screenshot shows the configuration for HiddenServiceDir and HiddenServicePort with the port 80 and the address 127.0.0.1:80. The right screenshot shows the same configuration but with the address changed to 127.0.0.1:5856. Both screenshots show the nano editor's status bar at the bottom with various keyboard shortcuts like Help, Zapisz, Wyszukaj, Cut, Execute, Wyjdź, Wczyt.plik, Zastąp, Paste, and Wyjustuj.

Przed/po

Po zapisaniu pliku i opuszczeniu edytora pora należy uruchomić usługę Tor:

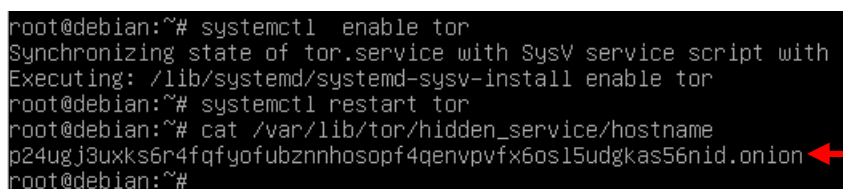
```
systemctl enable tor ↵
systemctl restart tor ↵
```

5. Na tym etapie usługa ukryta powinna już działać. Jaki jest adres usługi? - należy wyświetlić na konsoli zawartość pliku **hostname** z katalogu określonego w parametrze HiddenServiceDir. Na przykład poleceniem:

```
cat /var/lib/tor/hidden_service/hostname ↵
```

**Zrzucić ekran z tym adresem. Adres zapisać - do sprawozdania i do dalszego punktu.**

Alternatywnie plik z adresem można przesłać w inny sposób - np. załadować na konto shellowe po ssh (serwer sendzimir.metal.agh.edu.pl - konto z przedmiotu Systemy Operacyjne / i Administracja) lub użyć narzędzia typu wormhole (nie jest zainstalowane na kliencie ani na serwerze).



The image shows a terminal window with the following commands and output:  
root@debian:~# systemctl enable tor  
Synchronizing state of tor.service with SysV service script with /usr/sbin/systemd-install enable tor  
Executing: /lib/systemd/systemd-sysv-install enable tor  
root@debian:~# systemctl restart tor  
root@debian:~# cat /var/lib/tor/hidden\_service/hostname  
p24ugj3uuxks6r4fqfyofubznnhosopf4qenvpvfx6osl5udgkas56nid.onion  
root@debian:~#

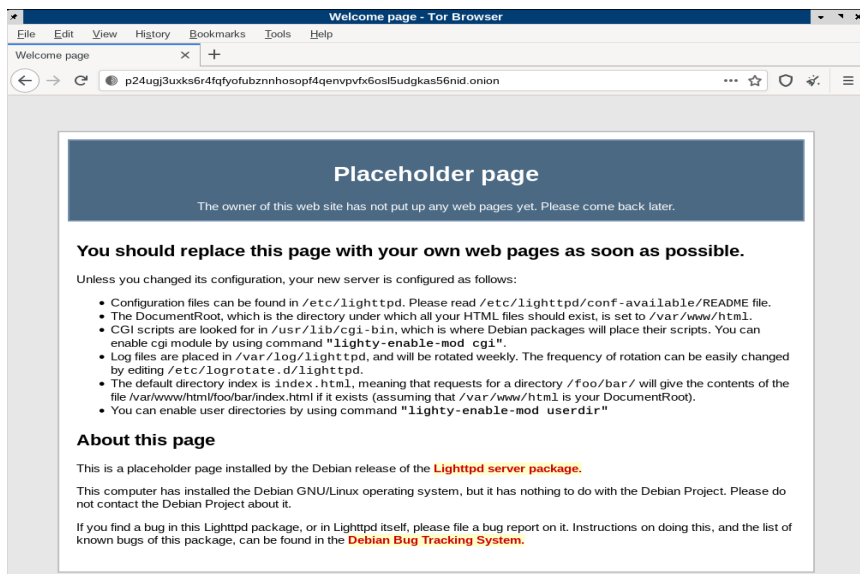
A red arrow points to the output of the cat command, which is the hidden service address: p24ugj3uuxks6r4fqfyofubznnhosopf4qenvpvfx6osl5udgkas56nid.onion.

6. Na drugim komputerze w pracowni, na którym działa **Tor browser**, wpisać adres i spróbować dostać się do usługi. Powinno dać się uzyskać domyślną stronę startową serwera lighttpd.

**Zmodyfikować /var/www/html/index.lighttpd.html** tak, by pod „Placeholder Page” dopisać:

**Bieżącą datę, Państwa imiona i pierwsze litery nazwisk.** Całych nazwisk proszę nie umieszczać.

Zrzut ekranu z tak zmodyfikowaną stroną powinien znaleźć się w sprawozdaniu.



Podobny zrzut, z komputera w pracowni lub własnego laptopa, powinien znaleźć się w sprawozdaniu.

7. Na serwerze w katalogu `/var/www/html` wygenerować plik testowy o rozmiarze podanym przez Prowadzącego laboratorium. W katalogu `/var/www/html` wydać polecenie np:

```
dd if=/dev/random of=./test.bin bs=30M count=1 ↵
```

gdzie **30M** to rozmiar w MB, a count to zadana przez Prowadzącego ilość bloków. W ten sposób np. 45M to 45MB, itp.

**PRZED POTWIERDZENIEM UPEWNIĆ SIĘ, ŻE NIE NADPISUJEMY SOBIE TWARDEGO DYSKU.** Ta kropka na początku „./test.bin” jest istotna. Poza `/dev/random` nie ma wpisanych żadnych urządzeń. Niewłaściwie użyte **dd nadpisuje bez ostrzeżenia.**

Zrobić zrzut ekranu zawierający polecenie oraz wynik `ls -l` w tym katalogu.

11. Instalujemy program do monitorowania łącza:

```
apt-get install iftop ↵
```

Uruchamiamy go poleceniem **iftop**. Możliwe jest zaobserwowanie nawiązywanych połączeń niemal „na żywo”. Trzy wartości szybkości wysyłania (rx) to średnie wartości z 2, 10 i 40 sekund.

12. Przeprowadzić analogiczne do punktu z pobieraniem pomiary prędkości pobierając nowo utworzony plik (w przeglądarce Tor na drugim komputerze, adres `http://[.....].onion/test.bin`). Dodatkowo sprawdzić wykorzystanie łącza na serwerze iftop-em przy 3 różnych trasach. Trasy (zrzuty ekranu z Tor Browser) również powinny znaleźć się w sprawozdaniu.

Po pomiarach, na wciąż działającym serwerze uruchamiamy program **htop**, robimy zrzut ekranu i notujemy użycie pamięci. Pobieramy plik jeszcze raz i notujemy użycie CPU.

Dodatkowo: Jeżeli w zespole ktoś posiada alternatywne łącze do Internetu (laptop z połączeniem Wifi, komórka z funkcją Tethering) można przeprowadzić analogiczne pomiary z innego łącza.

Istotne pytania, na jakie należy sobie odpowiedzieć pisząc wnioski:

- Zakładając ukrytą usługę - jak charakterystyka sieci Tor wpływa na to co umieszczamy? Co można umieszczać, a co sprawi problemy?
- Jakimi parametrami powinien odznaczać się serwer?
- Co robić, a czego nie robić na serwerze ukrytej usługi?

### **Sprzątanie po wykonanym laboratorium:**

- Usunąć katalog z Tor Browser z komputera klienckiego
- Usunąć w VirtualBox maszynę wirtualną z opcją „Usuń wszystkie pliki”.
- Usunąć plik .ova by nie marnował niepotrzebnie quoty.