pasja-informatyki.pl

Sieci komputerowe

Protokoły warstwy aplikacji

Damian Stelmach

Spis treści

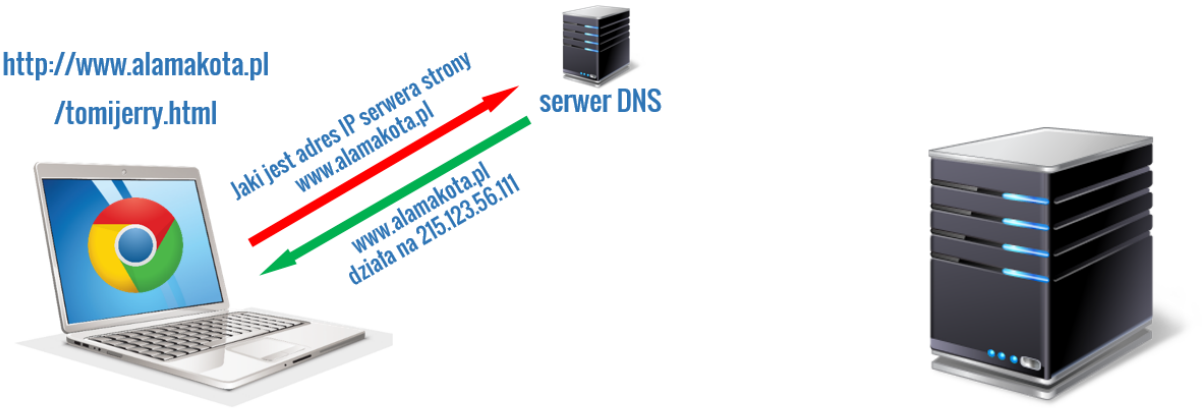
Kiedy uruchamiamy przeglądarkę internetową lub komunikator czy też program do wymiany plików, aplikacje te tworzą **interfejs komunikacyjny** pomiędzy siecią komputerową, a użytkownikiem. Oczywiście sama aplikacja, sam program komputerowy nie wystarczą do sprawnej komunikacji, bo do tego potrzebne są jeszcze wspomniane protokoły komunikacyjne, ale te są w tych programach zaimplementowane. Przykładowy protokół **warstwy aplikacji**, chyba jeden z najpopularniejszych czyli **HTTP** jest zaimplementowany w przeglądarce internetowej, podobnie jest ze wszystkimi komunikatorami oraz innymi programami wykorzystującymi komunikację sieciową, również one **zaimplementowane mają odpowiednie protokoły**.

Kiedy wpiszemy w przeglądarce adres strony internetowej, tak zwany adres **URL** (z ang. **Uniform Resource Locator**) to po wciśnięciu klawisza Enter, nasza przeglądarka nawiązuje połączenie z serwerem, na który dana strona jest przechowywana i żąda od niego określonego zasobu - najczęściej pliku zawierającego treść strony. Jeśli serwer posiada żądany zasób, przesyła jego zawartość do przeglądarki, ta interpretuje kod **HTML**, w którym strona została napisana i wyświetla użytkownikowi jej zawartość. Widać to na zmieszczonej poniżej grafice.

Oczywiście tak to wygląd w telegraficznym skrócie. W rzeczywistości proces ten jest nieco bardziej złożony. Weźmy sobie przykładowy adres internetowy:

http://www.alamakota.pl/tomijerry.html

Po jego wpisaniu i zatwierdzeniu, najpierw przeglądarka sprawdza **rodzaj protokołu**, następnie nazwę **domeny internetowej**, a na końcu dopiero brana jest pod uwagę nazwa **konkretnego pliku**. Później nasza przeglądarka odwołuje się do serwera **DNS**, w celu zamiany nazwy mnemonicznej, czyli alamakota.pl na **adres IP serwera** na którym ta strona jest przechowywana.



Znając już ten adres przeglądarka wysyła **żądanie** do serwera o udostępnienie pliku **tomijerry.html** znajdującego się w domenie **alamakota.pl**. Jeśli serwer dany zasób posiada, w odpowiedzi przesyła stosowny komunikat wraz z zawartością żądanego pliku. Zawartość tego pliku, czyli **kod HTML** jest przez przeglądarkę interpretowany i wyświetlany jako **strona WWW**.

Protokół **HTTP** standardowo działa na porcie **80** i definiuje kilka podstawowych rodzajów wiadomości czyli **żądań**, za pomocą których klient komunikuje się z **serwerem WWW**, do najważniejszych z nich należą żądania: **GET** oraz **POST**.

Metoda GET

GET służy do żądania od serwera danej strony WWW. Jego nagłówek wygląda mniej więcej tak:

```
GET /tomijerry.html HTTP/1.1
```

Zwiera oprócz nazwy żądanego zasobu, również stosowaną **wersję protokołu**. Gdy serwer taką wiadomość, takie żądanie odbierze, odpowiada klientowi stosownym komunikatem, jego nagłówek widoczny jest poniżej:

```
HTTP/1.1 200 OK /tomijerry.html
```

oraz żądanym zasobem. W żądaniu **GET** znajdują się jeszcze takie informacje jak: **nazwa hosta** (np. wp.pl), **nazwa przeglądarki**, z jakiej zostało wysłane żądanie, akceptowane przez przeglądarkę **typy plików**, preferowany **język strony** czy **kodowanie znaków**. W odpowiedzi serwera, znajdują się informacje, takie jak: **czas serwera**, nazwa **aplikacji serwera** (np. APACHE) czy **czas wygaśnięcia dokumentu**.

Jeśli z jakiś przyczyn nie może odesłać zasobu, odsyła **komunikat błędu**, np. **404**, który informuje, że żądany zasób **nie został znaleziony**, lub **403** informujący o **zabronionym dostępie do zasobów**. Wybrane kody komunikatów i błędów widoczne są w poniższych tabelach.

Kody informacyjne:

KOD	OPIS	ZNACZENIE
100	Continue	Prośba o dalsze wysyłanie zapytania
101	Switching Protocols	Zmiana protokołu
110	Connection Timed Out	Przekroczono czas połączenia. Serwer zbyt długo nie odpowiada.
111	Connection refused	Serwer odrzucił połączenie

Kody powodzenia:

KOD	OPIS	ZNACZENIE
200	OK	Odesłano zawartość żądanego dokumentu
201	Created	Wysłany dokument został zapisany na serwerze
202	Accepted	Zapytanie zostało przyjęte do obsłużenia, lecz jego zrealizowanie jeszcze się nie skończyło
204	No content	Brak zawartości – serwer zrealizował zapytanie klienta i nie zwraca żadnej treści
205	Reset Content	Serwer zrealizował zapytanie i klient powinien przywrócić pierwotny wygląd dokumentu

Kody błędów klienta:

KOD	OPIS	ZNACZENIE
400	Bad Request	Żądanie nie może być obsłużone przez serwer z powodu błędu klienta
401	Unauthorized	Żądanie zasobu, który wymaga uwierzytelnienia
403	Forbidden	Serwer zrozumiał zapytanie lecz konfiguracja bezpieczeństwa zabrania mu zwrócić żądany zasób
404	Not Found	Serwer nie odnalazł zasobu według podanego URL
405	Method Not Allowed	Metoda zawarta w żądaniu nie jest dozwolona dla wskazanego zasobu
406	Not Acceptable	Zażądany zasób nie jest w stanie zwrócić odpowiedzi mogącej, która może być obsłużona przez klienta
407	Proxy Authentication Required	Wymagane uwierzytelnienie do serwera pośredniczącego
408	Request Timeout	Koniec czasu oczekiwania na żądanie – klient nie przesłał zapytania do serwera w określonym czasie
409	Conflict	Żądanie nie może być zrealizowane, ponieważ występuje konflikt z obecnym statusem zasobu
411	Length required	Wymagana długość – serwer odmawia zrealizowania zapytania ze względu na brak nagłówka `Content-Length` w zapytaniu
415	Unsupported Media Type	Nieznany sposób żądania – serwer odmawia przyjęcia zapytania, ponieważ jego składnia jest niezrozumiała dla serwera

Kody błędów serwera:

KOD	OPIS	ZNACZENIE
500	Internal Server Error	Wewnętrzny błąd serwera – serwer napotkał problemy, które uniemożliwiły zrealizowanie żądania
501	Not Implemented	Serwer nie dysponuje funkcjonalnością wymaganą w zapytaniu
502	Bad Gateway	Błąd bramy – serwer – spełniający rolę bramy lub pośrednika – otrzymał niepoprawną odpowiedź od serwera nadrzędnego i nie jest w stanie zrealizować żądania klienta
503	Service Unavailable	Usługa niedostępna – serwer nie jest w stanie w danej chwili zrealizować zapytania klienta ze względu na przeciążenie
504	Gateway Timeout	Przekroczony czas bramy – serwer – spełniający rolę bramy lub pośrednika – nie otrzymał w ustalonym czasie odpowiedzi od wskazanego serwera HTTP, FTP, LDAP itp. lub serwer DNS jest potrzebny do obsłużenia zapytania
505	HTTP Version Not Supported	Nieobsługiwana wersja HTTP – serwer nie obsługuje bądź odmawia obsługi wskazanej przez klienta wersji HTTP

Metoda POST

Kolejny typ wiadomości to wiadomość **POST**, która służy do **przesyłania danych** na serwer. Kiedy storna internetowa zawiera np. **formularz** wysyłający dane na serwer, np. formularz rejestracji, to dane które w nim umieścimy wysyłane są właśnie za pomocą wiadomości **POST**.

Protokół HTTP mimo, że bardzo popularny, chyba najczęściej stosowany ze wszystkich protokołów warstwy aplikacji, nie jest bezpieczny. Metoda POST przesyła dane do serwera **jawnym tekstem**. Kiedy uda się przechwycić transmisję pomiędzy klientem, a serwerem, można odczytać informacje jakie chcemy przesłać na serwer.

Jest to bardzo niebezpieczne dlatego obecnie, większość stron WWW, na których istnieje możliwość przesłania na serwer jakieś informacji, czyli np. na tych stronach gdzie konieczne jest logowanie, stosowany jest już protokół **HTTPS** szyfrujący komunikację pomiędzy klientem a serwerem, działa on na **porcie 443**.

Pozostałe typy wiadomości, które mogą być wysyłane przez klienta do serwera WWW to:

Żądanie	Opis
Delete	Żądanie usunięcia zasobu z serwera
Head	Żądanie zasobu od serwera w formie nagłówka
Link	Żądanie ustanowienia relacji między istniejącymi zasobami
Options	Żądanie od serwera identyfikacji obsługiwanych metod
Put	Żądanie odebrania przez serwer od klienta pliku
Trace	Żądanie zwrócenia przez serwer nagłówków wiadomości wysłanej od klienta

Poczta elektroniczna stosuje dwa współpracujące ze sobą protokoły warstwy aplikacji. Jeden służy do **wysyłania poczty** i jest to protokół **SMTP**, a drugi do **odbierania wiadomości** i jest nim **POP3**. Obecnie do odbierania poczty elektronicznej stosowany może być również protokół **IMAP**. Protokoły te ściśle powiązane są z aplikacjami, czyli procesami uruchomionymi zarówno na komputerze klienckim gdzie tworzona i odbierana jest wiadomość, jak również na serwerze. Procesy te to **MUA** (z ang. Mail User Agent), **MTA** (z ang. Mail Transfer Agent) oraz **MDA** (za ang. Mail Delivery Agent), Proces MUA działa na urządzeniu klienckim, natomiast pozostałe dwa na serwerach pocztowych.

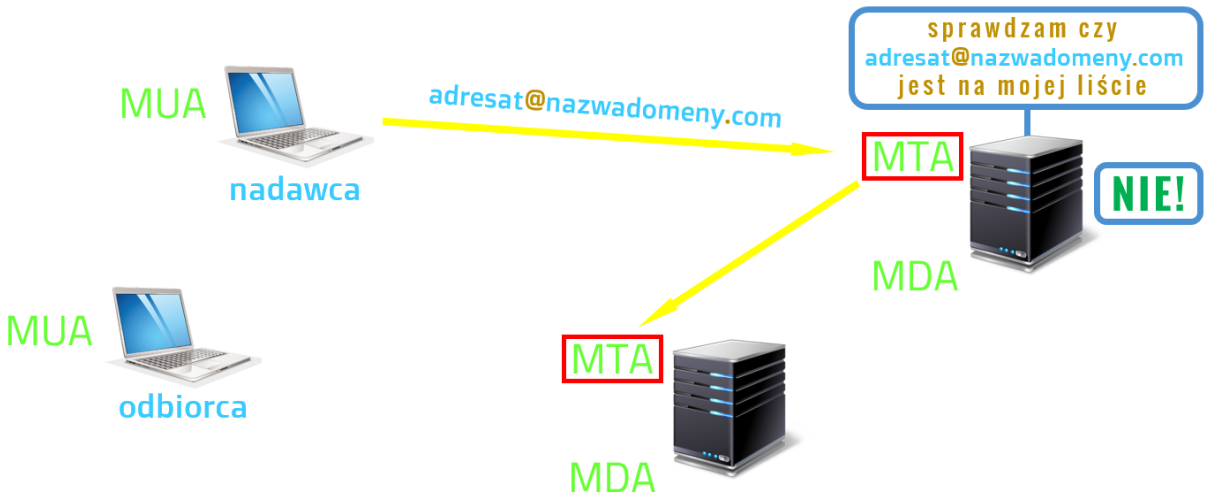
Uproszczony proces przesyłania wiadomości pocztowych z wykorzystaniem **Agentów** wygląda następująco:

1. Użytkownik tworzy wiadomość e-mail i za pomocą procesu **MUA** przekazuje ją do serwera poczty i procesu **MTA** działającego na tym serwerze.

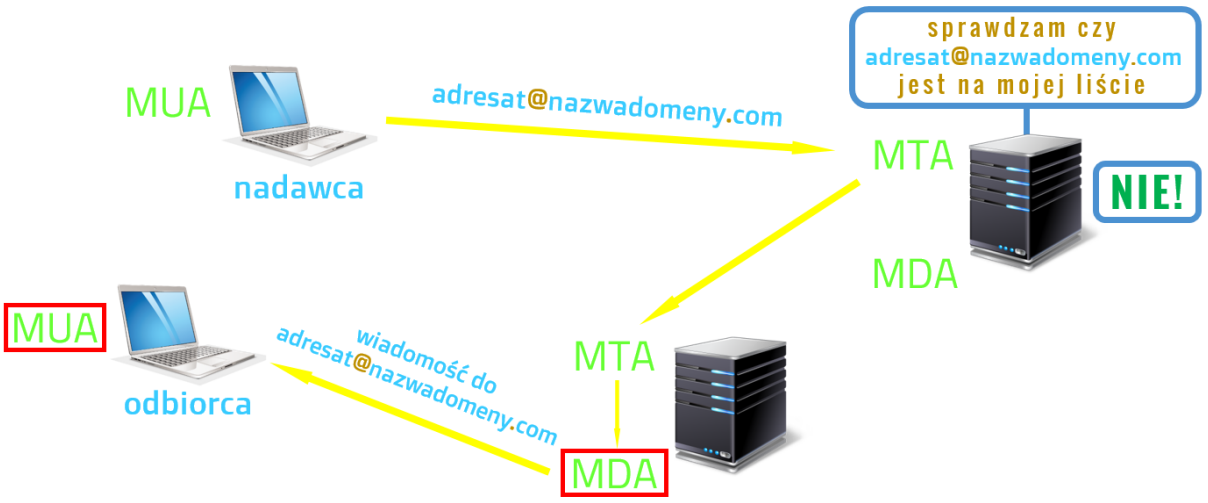
2. Proces ten analizuje nagłówek wiadomości, m.in. po to aby określić adresata wiadomości i sprawdza czy użytkownik do którego wiadomość jest kierowana znajduje się na jego liście użytkowników.

3. Jeśli tak jest, to przekazuje wiadomość do procesu **MDA**, który odpowiedzialny jest za dostarczenie jej do odpowiedniego adresata.

4. Jeśli adresat wiadomości **nie posiada konta** na tym serwerze to proces **MTA** dostarcza wiadomość do procesu **MTA** innego serwera, na którym konto tego użytkownika istnieje.



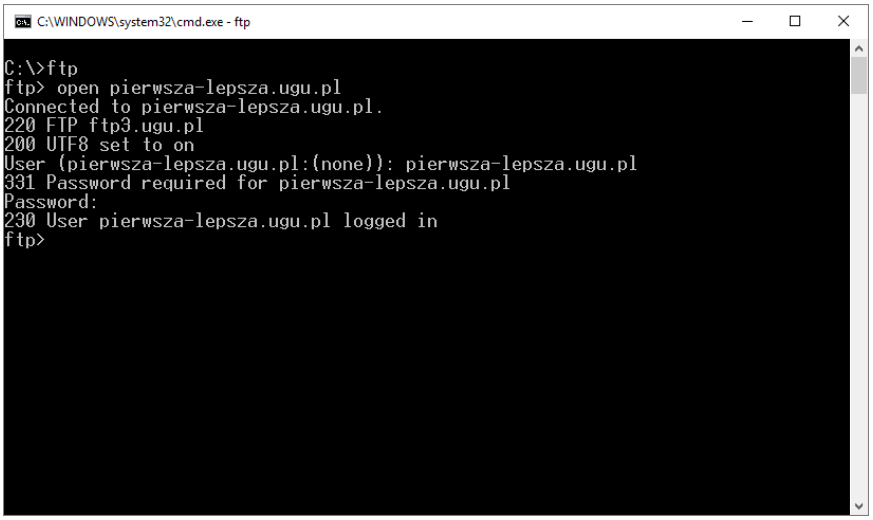
5. Serwer ten przekazuje wiadomość do procesu **MDA**, a ten dostarcza wiadomość do właściwego adresata.



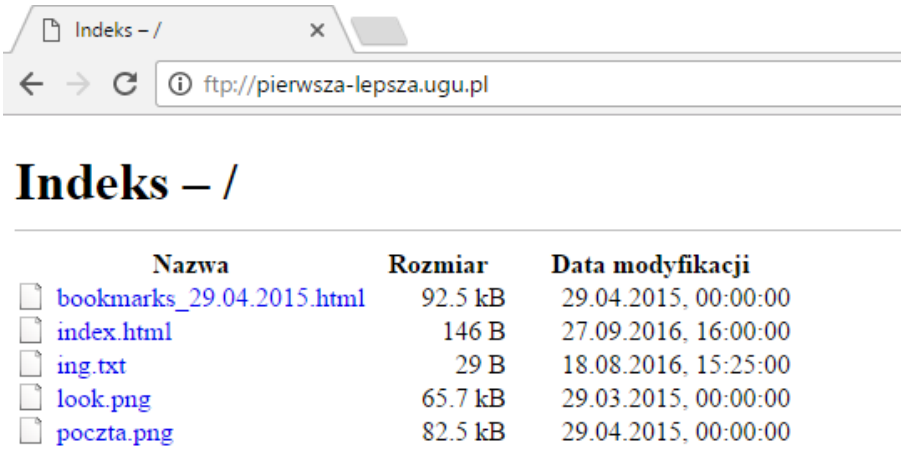
Porty, na których działają protokoły poczty elektronicznej widoczne są w tabeli poniżej.

Protokół	Numer portu
SMTP	**25**
POP3	**110**
IMAP	**143**
Szyfrowany SMTP	**465** lub **587**
Szyfrowany POP3	**995**
Szyfrowany IMAP	**993**

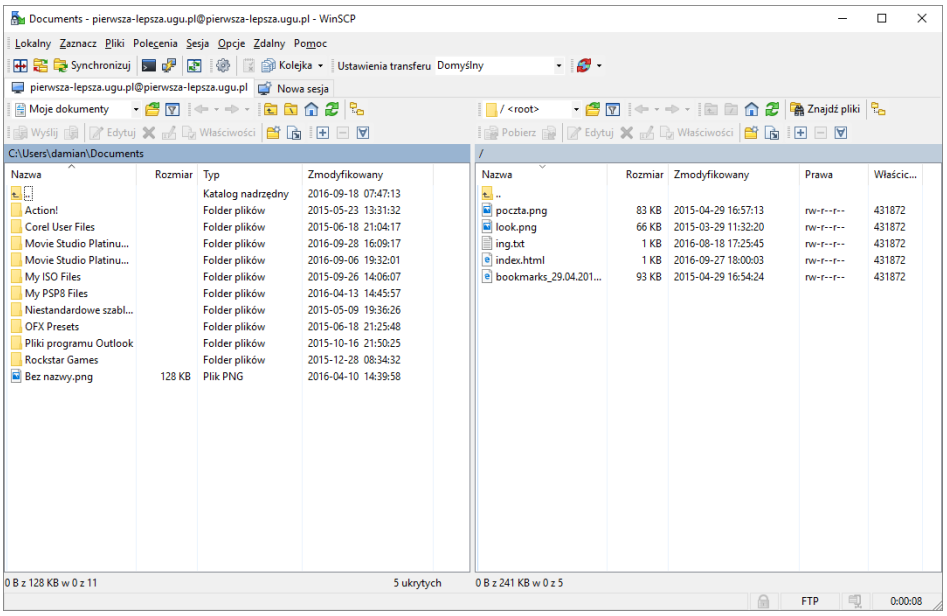
Trzecia już, nie mniej popularna usługa sieciowa to możliwość przesyłania i odbierania plików poprzez **FTP** (ang. **File Transfer Protocol**). Usługa ta, jednocześnie protokół komunikacyjny, jest bardzo często wykorzystywana kiedy chcemy wysłać **pliki strony internetowej**, na serwer WWW lub też kiedy chcemy po prostu wysłać jakieś pliki na serwer i **udostępnić** je innym użytkownikom. Aby wykonać operacje przesłania pliku na serwer czy też pobrania zasobu z serwera musimy skorzystać z **klienta FTP**, no i oczywiście na serwerze również taka usługa musi być uruchomiona. Klient FTP dostępny jest na każdym systemie operacyjnym i można z niego korzystać np. za pomocą wiersza poleceń, co jednak jest dość nie wygodne, ale możliwe.



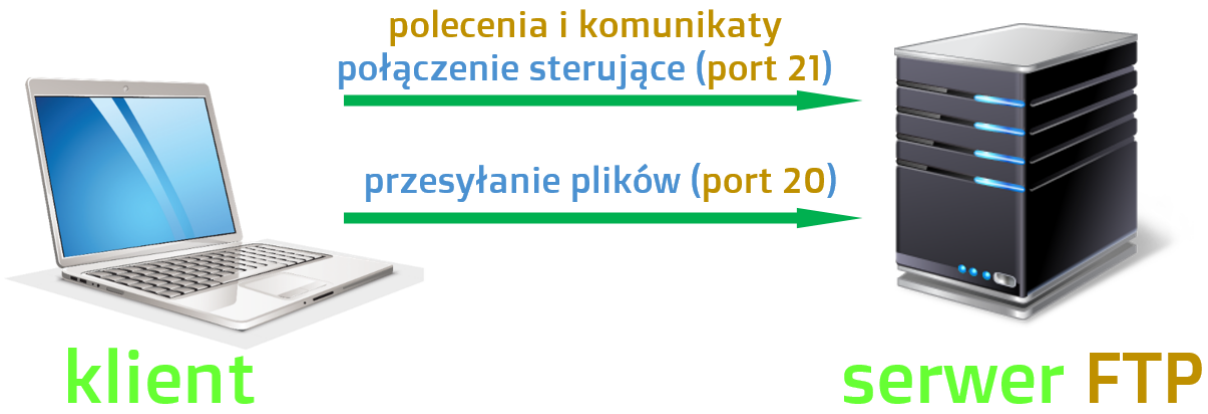
Jeśli korzystamy z FTP tylko do pobierania plików to śmiało możemy to zrobić za pomocą przeglądarki internetowej. Większość popularnych przeglądarek, o ile nie wszystkie mają wbudowanego **klienta FTP**.



Jeśli natomiast chcemy pliki na serwer przesyłać to warto skorzystać z dedykowanych programów, takich jak **FileZilla** czy też **WinSCP**, są one darmowe i spokojnie można je pobrać z sieci.



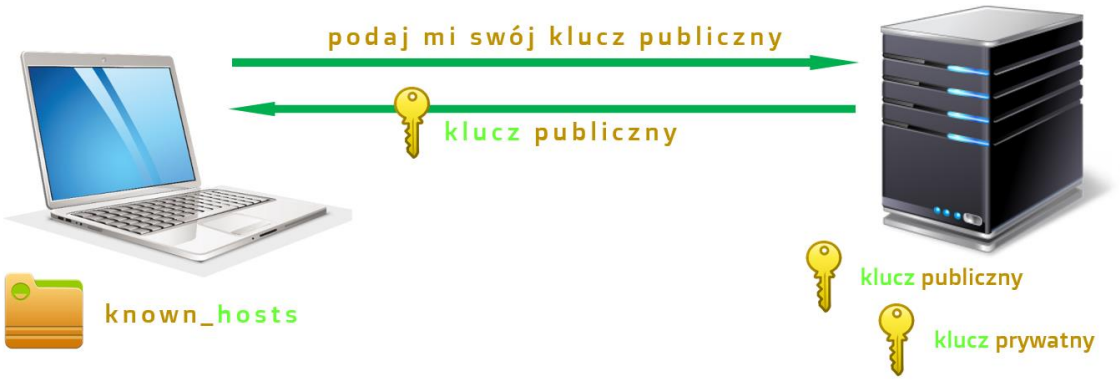
W przypadku tego protokołu, aby komunikacja mogła być zrealizowana we właściwy sposób należy zestawić **dwa połączenia** pomiędzy klientem a serwerem. Pierwsze połączenie służy tylko do wysyłania poleceń i komunikatów i nazywane jest połączeniem sterującym (działa na **porcie 21**), drugie natomiast, działające na **porcie 20** służy do przesyłania plików z i do serwera. Aby zabezpieczyć dostęp do serwera FTP stosuje się autoryzację użytkowników, dokładnie taką samą jak w przypadku logowania się do profilu na portalu społecznościowym czy do poczty elektronicznej, choć czasem, jeśli zasoby mają być dostępne dla większej liczby odbiorców, to realizuje się dostęp przez tak zwanego użytkownika anonimowego (ang. **Anonymous**), dzięki czemu niewymagana jest autoryzacja. Takie rozwiązanie powinno się stosować tylko wówczas, kiedy użytkownicy mogą pobierać dane z serwera. Upload plików, czyli umieszczania ich na serwerze zawsze dostępne jest tylko dla użytkowników posiadających login i hasło.



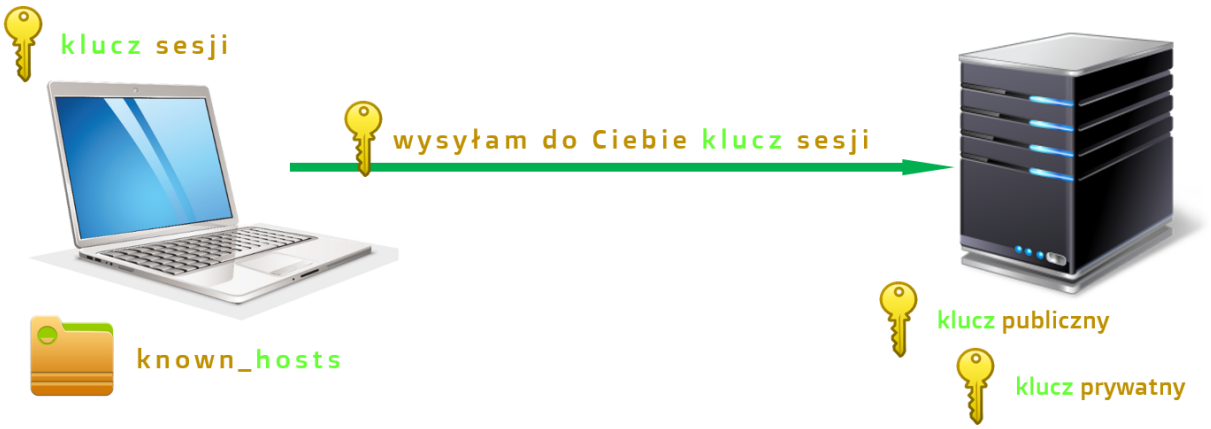
Kolejny często wykorzystywany protokół warstwy aplikacji, to protokół zdalnego zarządzania hostami, zwany **SSH** (ang. Secure Shell). Dla osób nie zajmujących się informatyką na co dzień, nazwa ta niewiele mówi, ponieważ nie jest to protokół, którego używają "zwykli zjadacze chleba", korzystający ze stron WWW czy też poczty elektronicznej. Stosowany jest on przez administratorów do **zarządzania serwerami** znajdującymi się bardzo często w różnych geograficznie miejscach, nie koniecznie w miejscu pracy. Stosują go też osoby, które mają np. wykupione **serwery VPS** i w ten sposób nimi administrują. Protokół ten wywodzi się z innego protokołu zdalnego dostępu, protokołu **TELNET** i jest jego, można powiedzieć lepszą wersją. Dlaczego? Dlatego, że TELNET, który notabene jest chyba najstarszym protokołem warstwy aplikacji, nie szyfruje komunikacji pomiędzy klientem a serwerem, komunikaty przesyłane są jawnym tekstem, a co za tym idzie istnieje możliwość przechwycenia komunikacji i podejrzenia jakie informacje są w sesji przesyłane. Obecnie jest to sytuacja nie do przyjęcia, dlatego też do zdalnego zarządzania hostami stosowany jest właśnie szyfrowany protokół SSH.

Domyślnym algorytmem szyfrowania komunikacji jest **algorytm RSA**, istnieje również możliwość szyfrowania danych za pomocą nieco słabszego **algorytmu DSA**. Podczas instalacji serwera SSH tworzona jest para kluczy - **klucz publiczny i prywatny** serwera - służą one do

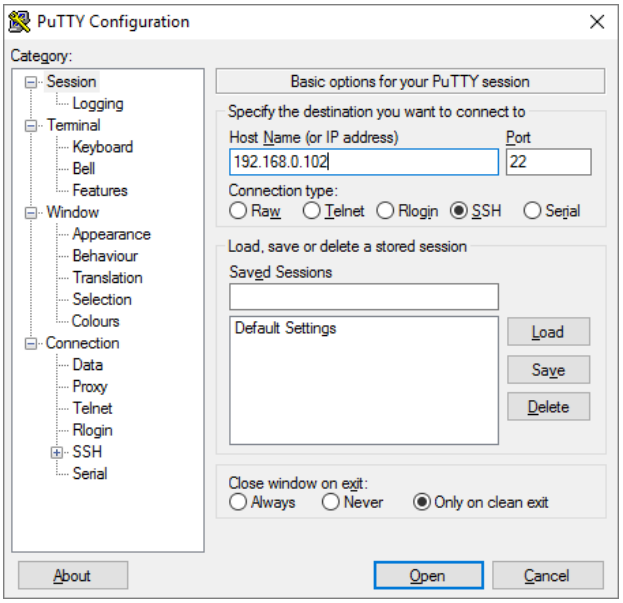
szyfrowania i deszyfrowania komunikacji. Podczas pierwszego połączenia z serwerem, klient, zapisuje publiczny klucz serwera na swoim dysku, w pliku **known_hosts**.



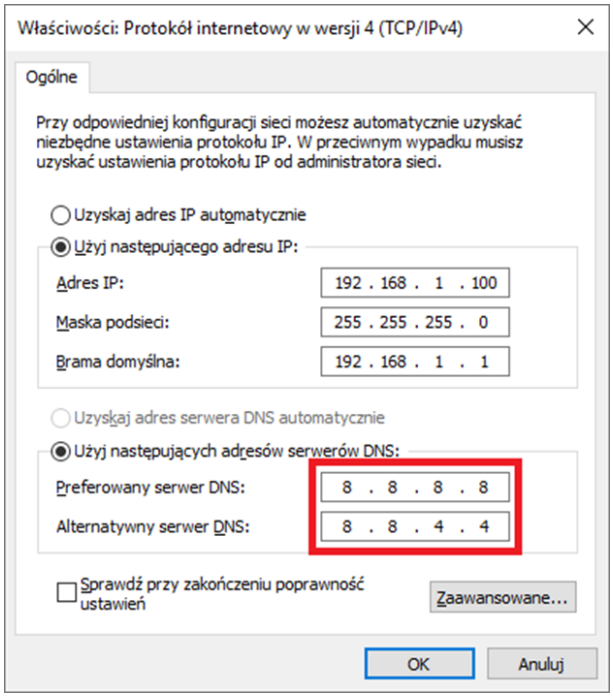
Następnie tworzy tak zwany **klucz sesji**, który będzie stosowany do szyfrowania całej komunikacji. Klucz sesji zostaje zaszyfrowany kluczem publicznym otrzymanym wcześniej od serwera i jest do niego odsyłany. Od tego momentu cała komunikacja szyfrowana jest kluczem sesji.



Standardowo SSH działa na **porcie 22**. Jednym z najpopularniejszych programów klienckich wykorzystujących SSH, jest program **PUTTY**, jest darmowy i można go pobrać z sieci, a co więcej nie wymaga instalacji. Aby połączyć się zdalnie z hostem, wystarczy go uruchomić, podać nazwę hosta lub jego adres IP, wybrać SSH jeśli domyślnie nie jest zaznaczony i kliknąć OPEN. Jeśli po raz pierwszy łączymy się ze zdalnym hostem potwierdzamy chęć nawiązania połączenia i już możemy zdalnie nim zarządzać.


PuTTY Configuration
Category:
Session
Logging
Terminal
Keyboard
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
Telnet
Rlogin
SSH
Serial
Basic options for your PuTTY session
Specify the destination you want to connect to
Host Name (or IP address)
Port
192.168.0.102
22
Connection type:
Raw
Telnet
Rlogin
SSH
Serial
Load, save or delete a stored session
Saved Sessions
Default Settings
Load
Save
Delete
Close window on exit:
Always
Never
Only on clean exit
About
Open
Cancel

DNS to protokół, usługa, zamieniająca **nazwy domenowe**, zrozumiałe dla człowieka na **adresy IP** urządzeń w sieci. Wyobraźmy sobie sytuacje, gdzie DNS nie istnieje, a my chcemy wyświetlić w przeglądarce naszą ulubioną stronę. Zamiast nazwy domeny, czyli adresu w postaci słownej musimy wpisać adres IP, np. taki: **212.56.93.112**. Dla większości z nas nie byłby to problem, można zapamiętać kilka cyfr. Natomiast w Internecie jest wiele stron WWW i zapamiętanie wielu adresów liczbowych byłoby już trudne. Co więcej w takim zapisie liczbowym, łatwo jest się pomylić, a w świecie Internetu, taka drobna pomyłka może skutkować tym, że wejdziemy na inna stronę WWW, niż zamierzaliśmy. To jest jedna strona medalu, druga jest taka, że adresy IP serwerów, może nie bardzo często, ale jednak się zmieniają. Kiedy nasza strona zmieni adres IP, a nie działa usługa DNS musimy uczyć się tego adresu na nowo, jeszcze raz go zapamiętać. DNS rozwiązuje za nas ten problem ponieważ on sam, w swojej **bazie rekordów** zmieni ten adres i przypisze go do nazwy domeny. Wtedy dla nas, użytkowników nie ma znaczenia pod jakim adresem IP strona się znajduje, ważne jest to, że znamy słownie jej adres, jej domenę, które to nie ulegają zmianie. DNS to usługa działająca w **architekturze klient - serwer**, jednak nie mamy tutaj do czynienia z klientem jako programem komputerowym takim jak przeglądarka czy program do wymiany plików. Na komputerze uruchamiana jest po prostu **usługa systemowa**, zwana z języka angielskiego **DNS resovler** i ona obsługuje wszystkie aplikacje na komputerze klienta, które wymagają zamiany nazw. Zawsze, kiedy konfigurujemy urządzenie sieciowe, czy też zwyczajnie komputer powinniśmy podawać **dwa adresy serwerów DNS**, tak aby w przypadku braku komunikacji z jednym z nich, drugi realizował funkcję zamiany nazw.



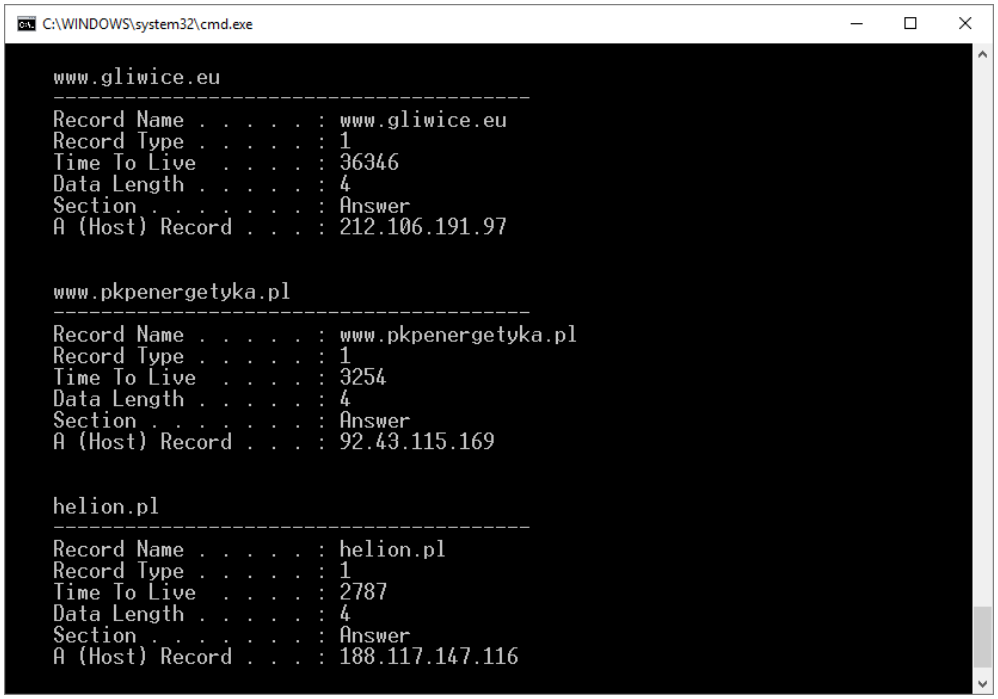
Serwer DNS przechowuje różne **typy rekordów**, m. in rekordy typy **A** i **AAAA** zawierające adresy urządzeń końcowych czy też rekordy typu **MX** obsługujące wymianę poczty elektronicznej, bo pamiętać należy, że DNS nie tylko zamienia adresy domenowe na adres IP dla stron WWW, ale także dla serwerów poczty elektronicznej. Zamiana nazw wygląda mnie więcej tak:

1. Klient wysyła żądanie do serwera DNS, a ten sprawdza, czy w swojej bazie posiada dany rekord.

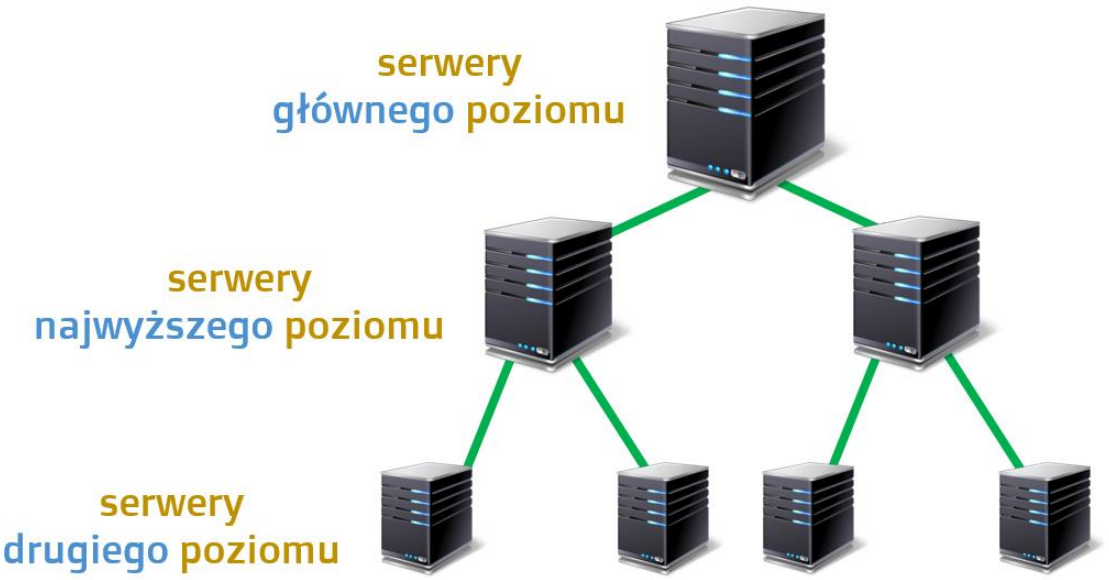
2. Jeśli tak przekształca nazwę na adres IP i odsyła do klienta.

3. Jeśli nie, to kontaktuje się z innymi serwerami, aby te dany rekord wyszukały w swoich bazach.

Wysyłanie zapytań serwera DNS, który nie znalazł rekordów w swojej bazie do innych serwerów może powodować nadmierny ruch sieciowy, co jest sytuacją nieporządną. Aby zapobiec nadmiernemu i niepotrzebnemu ruchowi w sieci, kiedy inny serwer odnajdzie dany rekord i prześle go do serwera przypisanego do naszego urządzenia, ten zapisuje sobie ten rekord w pamięci podręcznej, tak aby w przyszłości już nie musieć się odwoływać do innego serwera po ten sam adres. Zdecydowanie przyspiesza to potem zamianę nazw, ponieważ nasz serwer DNS nie szuka już rekordu po innych serwerach tylko od razu zamienia nazwy. Podobnie usługa DNS na komputerze osobistym przechowuje poprzednio przekształcone nazwy. Można to sprawdzić, wprowadzając na komputerze z systemem Windows polecenie **ipconfig/displaydns**. Zobaczymy wówczas jakie odwzorowania są zapisane w pamięci podręcznej usługi DNS naszego komputera.



Hierarchia serwerów DNS ta ma postać odwróconego drzewa, gdzie korzeń, czyli serwery DNS głównego poziomu znajdują się na samej górze. **Serwery głównego** poziomu przechowują informację jak dotrzeć do **serwerów najwyższego poziomu**, to z kolei przechowują informację jak dotrzeć do **serwerów drugiego poziomu** itd. Domeny najwyższego poziomu określają kraj lub też typ organizacji np. **.pl**; **.de**; czy **.uk** określające kraj, **.org** określa organizacje non profit, **.com** określa przedsiębiorstwo czy **.gov**, która to określa instytucje rządowe.

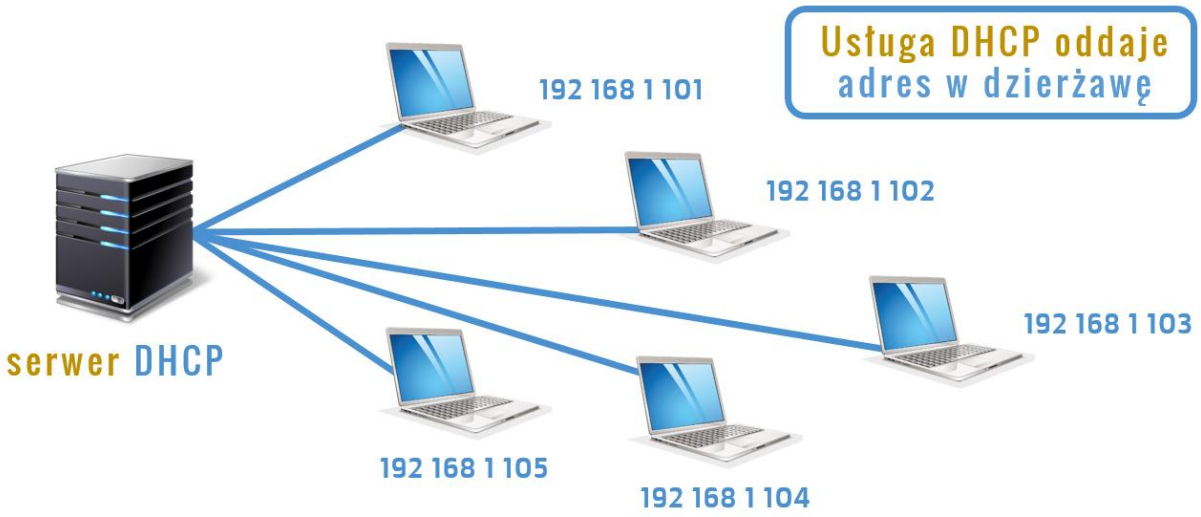


W takim, przykładowym adresie wyróżniamy domenę najwyższego poziomu, następnie domenę drugiego poziomu, i na koniec domenę 3 poziomu.

poczta.wp.pl

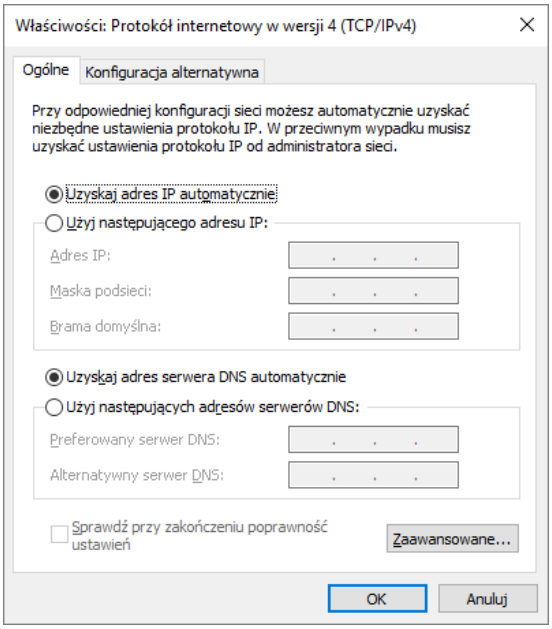
Oczywiście nie wszystkie adresy muszą zawierać, aż tyle poziomów domen, nie które zwierają tylko domeny najwyższego i drugiego poziomu, np. **wp.pl**, **pasja-informatyki.pl**, **szkola.pl**.

No i na koniec protokół **DHCP**. Podobnie jak omówiony przed chwilą DNS również jest to protokół działający jako usługa, a nie jako program czy aplikacja. DHCP umożliwia podłączonym do sieci komputerom pobieranie **adresu IP**, **maski podsieci**, **adresu bramy** i **serwera DNS** ze skonfigurowanej wcześniej **puli adresów**. Serwer DHCP może być skonfigurowany na osobnym komputerze i stanowił będzie osobne urządzenie w sieci przydzielające komputerom klienckim adresy IP, może również działać na już istniejącym serwerze jako osobna usługa, osobny proces. Obecnie również rutery, które mamy w domach pozawalają na skonfigurowanie takiej usługi. Przydzielanie adresów komputerom klienckim poprzez usługę DHCP, zwane **przydzielaniem dynamicznym** jest bardzo wygodnym rozwiązaniem dla administratorów, szczególnie w dużych sieciach, gdzie bardzo często pojawiają się nowe komputery i ich użytkownicy. W sieci, w której pracuje 100, 200 czy 500 komputerów, do tego sporo urządzeń przenośnych, sama konfiguracja adresów IP byłaby zajęciem uciążliwym, a przede wszystkim czasochłonnym. Oczywiście nie wszystkie urządzenia w sieci mogą uzyskiwać adresy w ten sposób ponieważ niektóre z nich, takie jak **serwery aplikacji**, **baz danych**, **uwierzytelniania użytkowników**, ale również **drukarki sieciowe** czy **rutery** powinny, a wręcz muszą posiadać adresy przydzielone statycznie, czyli ręcznie, dlaczego? Dlatego, że usługa DHCP, skonfigurowana na serwerze nie przydziela komputerom danego adresu IP na zawsze, na stałe. Ona tylko **oddaje w dzierżawę** taki adres, na czas ustalony podczas konfiguracji DHCP, może to być kilka godzin, kilka dni, ale nie na stałe, chodź i od tego są pewne wyjątki, ale o tym opowiem kiedy to będziemy konfigurować już konkretny serwer DHCP.



Komputer, który zostaje wyłączony, oddaje adres, który dzierżawił i ten adres wraca do puli. Inne urządzenie może wówczas ten adres wydzierżawić. Kiedy serwery, rutery czy drukarki

sieciowe takie adresy by dzierżawiły to po jakimś czasie prawdopodobnie musiałyby je zwrócić do puli i nie ma gwarancji, że ponownie uzyskałyby ten sam adres. Komputery klienckie komunikując się z jakimkolwiek serwerem czy innym ważnym urządzeniem działającym w sieci odwojują się do niego poprzez adres IP, jeśli ten często by się zmieniał, to pewne usługi dla użytkowników w sieci lokalnej mogłyby być przez pewien czas niedostępne, co szczególne w warunkach korporacyjnych jest nie do przyjęcia. Aby komputery z systemem Windows, pobierały adresy z serwera DHCP, w konfiguracji adresacji należy wybrać opcję „**Uzyskaj adres IP automatycznie**”.



Przedstawione powyżej protokoły warstwy aplikacji to tylko niewielka część z całej listy dostępnych protokołów tej warstwy. W sieciach komputerowych istnieje wiele innych usług, a każda z nich funkcjonuje na innym protokole. Trudno byłoby tutaj wymienić wszystkie, dlatego przedstawione zostały te najpopularniejsze i najczęściej wykorzystywane. Dla osób zainteresowanych zgłębianiem tematyki protokołów komunikacyjnych warstwy aplikacji odsyłam do literatury fachowej. W poniższej tabeli przedstawiłem zbiór popularnych protokołów warstwy aplikacji wraz z numerami portów. Z pewnością przydadzą się do powtórki przed sprawdzianem czy egzaminem zawodowym.

Protokół	Opis	Numer portu
HTTP	Protokół przesyłania dokumentów hipertekstowych (stron WWW)	**80**
HTTPS	Szyfrowany protokół HTTP wykorzystujący protokoły szyfrujące TLS lub SSL	**443**
POP3	Protokół odbierania poczty elektronicznej	**110** (szyfrowany **995**)
IMAP	Protokół odbierania poczty elektronicznej umożliwiający zarządzanie folderami znajdującymi się w skrzynce pocztowej	**143** (szyfrowany **993**)
SMTP	Protokół wysyłania poczty elektronicznej	**25** (szyfrowany **465** lub **587**)
FTP	Protokół przesyłania plików	**21** (polecenia) i **20** (pliki)
FTPS	Szyfrowany protokół FTP	**990**
TELNET	Protokół połączenia terminalowego	**23**
SSH	Szyfrowany protokół połączenia terminalowego	**22**
DNS	Protokół zamiany nazw domenowych na adresy IP	**53**
DHCP	Protokół automatycznej konfiguracji hostów w sieci	**67** i **68** (dla IPv6 **546** i **547**)
LDAP	Protokół usług katalogowych (np. Active	**389** (szyfrowany **639**)

	Directory w Windows Server)	
SNMP	Protokół konfiguracji urządzeń sieciowych	**161**
MySQL	System zarządzania bazami danych	**3306**
PostgreSQL	System zarządzania bazami danych	**5432**