HTML, XHTML, DHTML

Język HTML (HyperText Markup Language) jest oficjalnym formatem, jak również językiem opisu dokumentów WWW zawierających hiperłącza[1] do innych stron WWW. Został on powołany do życia w roku 1989 przez Tim'a Berners-Lee i Anders'a Berglund. Pierwsza wersja HTML'a była bardzo prymitywna – pozwalała jedynie na prostą hierarchiczną prezentacj ę informacji tekstowych. W kolejnych odsłonach specyfikacji standardu HTML dodano obsługę czcionek, tabelek, zamieszczania obiektów multimedialnych. Nie jest on językiem programowania, lecz zbiorem reguł umożliwiających formatowanie dokumentów WWW.

Opisując dokumenty hipertekstowe wymagane jest przestrzeganie pewnych reguł. Dzięki HTML można zapisać informacje dotyczące wyglądu strony. Na podstawie tego opisu, zawartego w dokumencie WWW, przeglądarka może odtworzyć wygląd strony WWW i wyświetlić ją na ekranie. Dokument HTML jest zwykłym plikiem tekstowym, w którym znajdują się polecenia HTML. Przy jego tworzeniu korzysta się z opisowych identyfikatorów (znaczników) służących do rozróżniania poszczególnych części w dokumencie. Znaczników używa się do dzielenia dokumentu na logiczne części (m.in. tytuły, akapity, tabele). Znaczniki po pierwsze określaj ą strukturę dokumentów (aby przeglądarka mogła go wyświetlić), po drugie, określają odsyłacze, które kierują programy do istotnych części dokumentu. Wszystkie znaczniki zaczynają się znakiem „mniejsze niż" („<") i kończą znakiem „większe niż" („>").

Elementy zamykające są podobne do elementów otwierających, z tą różnicą, że w elementach zamykających, tuż za znakiem „<" znajduje się prawy ukośnik „/"(ang. slash). Cały dokument powinien być objęty parą znaczników <HTML> </HTML>, definiujących początek dokumenty. Między nimi powinna znaleźć się para znaczników <HEAD> </HEAD>, które definiują nagłówek i elementy opisujące dokument HTML. W części „HEAD” należy umieścić element <TITLE></TITLE>, który precyzuje tytuł strony, wyświetlany na pasku tytułowym przeglądarki. Ważną rzeczą, która także powinna znaleźć się w części nagłówka jest informacja o stronie kodowej dokumentu. Strony kodowe są standaryzowane przez organizację ISO. Dla Polski przewidziany jest standard ISO-8859-2, który stał się także Polską Nomą. Jeżeli chcemy, aby polskie litery diakrytyczne były poprawnie wyświetlane, powinniśmy tworzyć dokument w tym standardzie. Pozostałe informacje powinny być objęte z kolei znacznikami <BODY> </BODY>, które określają „ciało” strony WWW, innymi słowy zawartość. Przykładowo do oznaczenia w dokumencie HTML tytułu najwyższego stopnia używa się elementu H1. Przykład:

```
<HTML>

<HEAD>

<meta http-equiv=”content-type” content=”text/html; charset=iso-8859-2″>

<TITLE> Tytuł strony </TITLE>

</HEAD >

<BODY>

<H1> Tytuł </H1>

</BODY>

</HTML>
```

Jak widać znaczniki są tekstami ASCII. Jest to cecha wszystkich elementów HTML. Z tego też względu dokument HTML można utworzyć za pomocą najprostszego edytora tekstu, ręcznie dodając znaczniki. Metoda taka jest dość męcząca, dlatego lepiej skorzystać z dostępnych na rynku edytorów, które pomagają (m.in. poprzez kolorowanie składni) tworzyć takie dokumenty. [12][13][15]

DHTML, czyli Dynamic Hypertext Markup Language (dynamiczny hipertekstowy język znaczników) to rozszerzenie języka HTML o nowe możliwości: dokładne określenie położenia obiektów (pozycjonowanie) i wynikającą z tego możliwość animacji, użycie dowolnych, ładowanych z Sieci typów czcionek, pełną interakcje, zmiany „w locie” zawartości strony i stylów oraz tzw. filtry i przenikania. Wszystkie te możliwości całkowicie integruj ą się z HTML-em. DHTML nie powinien być rozumiany jako nowa, odrębna technologia. W rzeczywistości to zespół (nawet nie do końca ze sobą powiązanych) rozszerzeń, dodających pewne nowe właściwości poszczególnym elementom strony.[14]

Dynamicznym HTML-em bardzo często mylnie nazywa się technologie umożliwiające dynamiczne generowanie dynamicznych stron WWW. W skład DHTML wchodzą technologie takie jak (X)HTML, DOM, CSS, SVG (i inne aplikacje XML-a), rozszerzenia przeglądarek (np. w postaci filtrów) i JavaScript (lub inny język skryptowy działający po stronie przeglądarki stron WWW), który jest najistotniejszy, ponieważ dzięki niemu można „wprawić w ruch” pozostałe technologie. Po raz pierwszy, w czerwcu 1997r., obsługa DHTML została zaimplementowana w przeglądarce Netscape Navigator 4, aczkolwiek obsługa DOM była bardzo ograniczona, przez co tworzenie skryptów nastręczało licznych trudności.

W październiku 1997 Internet Explorer został wzbogacony o obsługę DHTML. W przeciwieństwie do Netscape Navigator’a, była to bardzo elastyczna implementacja DOM.

Niestety obie implementacje nie były ze sobą zgodne. W październiku 1998 roku W3C[2] ogłosiło ostateczną i oficjalną wersję standardu DOM (zbliżony do wersji Microsoftu), co umożliwiało opracowanie przeglądarki, która obsługiwałaby DHTML na podstawie oficjalnych standardów.

XHTML w języku angielskim oznacza Extensible HyperText Markup

Language co w języku polskim brzmi: „rozszerzalny hipertekstowy język znaczników”. Jest to aplikacja XML. Służąca do tworzenia stron WWW ogólnego przeznaczenia. XHTML jest następcą nierozwijanego już języka HTML, którego specyfikacje przygotowuje organizacja W3C. W odróżnieniu od HTML-a, dokumenty pisane w XHTML są zgodne ze specyfikacją XML. Atutem XHTML jest możliwość łączenia z innymi językami zgodnymi z XML, np. MathML czy SVG. Obecnie nowe przeglądarki, takie jak Firefox czy Opera, praktycznie w pełni obsługują XHTML, lecz przeglądarka mająca ciągle największy udział w rynku – Internet Explorer – w ogóle nie obsługuje XHTML-owego typu zawartości. W praktyce zmusza to webmasterów do stosowania dla dokumentów XHTML starego HTML-owego typu zawartości – dzięki temu, że XHTML w wersji 1.0 „symuluje” HTML 4 (tzn. posiada praktycznie taki sam zestaw elementów i atrybutów), wyświetlanie XHTML jako HTML nie sprawia większych problemów w żadnej przeglądarce. Jednak obecnie coraz częściej wykorzystuje się metodę negocjowania zawartości do prezentowania XHTML-a w Sieci.[13]

---

[1] Pewien fragment dokumenty opisanego w języku HTML. Jest tekstem zawierającym łącze do informacji przechowywanej w sieci WWW. Najczęściej jest to łącze do innej strony WWW. Do oznaczenia łącza stosuje się zwykle pogrubienie lub podkreślenie, jak również można zastosować inny kolor liter.

[2] World Wide Web Consortium – organizacja zajmująca się ustanawianiem standardów pisania i przesyłu stron WWW. http://www.w3.org/

[13] „Html- to proste ”, McBride P.K, McBride N, Warszawa: ReadMe 2001.

[14] „DHTML – Dynamiczny staruszek’, Radosław Tereszczuk, http ://www. cyb er.com.pl/archiwum/9/7. shtml

Jeśli szukają Państwo pomocy w napisaniu własnej pracy - potrzebują Państwo fachowych konsultacji to polecamy stronę pisanie prac - profesjonalna pomoc w pisaniu prac w granicach prawa.

Repeater (Regenerator)

podrozdział z pracy o sieciach komputerowych

Repeater (znany również jako **regenerator**) to urządzenie sieciowe używane w telekomunikacji i sieciach komputerowych w celu rozszerzenia zasięgu sygnału. Działa poprzez odbiór osłabionego lub zdegradowanego sygnału, jego wzmocnienie lub regenerację, a następnie retransmisję do miejsca docelowego. Proces ten pomaga utrzymać integralność sygnału na dużych odległościach, gdzie sygnał mógłby stać się zbyt słaby, by był odbierany w sposób czytelny.

Repeater jest prostym urządzeniem pomocniczym, regenerującym sygnał przesyłany kablem, co pozwala na zwiększenie długości połączenia, a co za tym idzie – zwiększenie rozpiętości sieci. *Repeater* nie zmienia w żaden sposób struktury sygnału, poza jego wzmocnieniem. *Repeater* jest nieinteligentnym (*dumb*) urządzeniem, które charakteryzuje się następującymi cechami:

1. używany jest głównie w liniowych systemach kablowych;
2. działa na najniższym poziomie stosu protokołów – na poziomie fizycznym;
3. dwa segmenty sieci, połączone za pomocą *repeater'a*, muszą używać tej samej metody dostępu do medium;
4. segmenty sieci połączone za pomocą *repeater'a* stają się częścią tej samej sieci i mają te same adresy sieciowe

(węzły w segmentach rozszerzających sieć muszą mieć różne adresy od węzłów w segmentach istniejących);
5. przekazują pakiety z prędkością transmisji w sieci;

W *repeater'ach* należy raczej widzieć urządzenia, które służą do przyłączenia do sieci stacji dalej położonych, niż urządzenia pozwalające na zwiększenie liczby stacji w sieci.

Repeater działa na warstwie fizycznej (warstwa 1) modelu OSI. Jest powszechnie wykorzystywany w sieciach przewodowych i bezprzewodowych, szczególnie w sytuacjach, gdy odległość transmisji przekracza limity medium, takie jak w sieciach Ethernet, światłowodowych lub w komunikacji radiowej.

Repeater pełni funkcje polegające na wzmocnieniu sygnału oraz jego regeneracji, dzięki czemu sygnał może pokonać większą odległość. Jest to szczególnie przydatne w systemach przewodowych, takich jak kable miedziane (np. kable koncentryczne) lub światłowody, gdzie sygnał może ulegać tłumieniu na dużych odległościach. W sieciach cyfrowych regenerator oprócz wzmocnienia sygnału, odbudowuje go, eliminując szumy i zniekształcenia, które mogły pojawić się podczas transmisji. Dzięki temu zapewnia, że sygnał pozostaje silny i czytelny.

Repeater znajduje zastosowanie w różnych typach sieci, w tym w sieciach lokalnych (LAN), rozległych (WAN) oraz w sieciach bezprzewodowych. Jest niezbędny do łączenia sieci na dużych obszarach, na przykład w przypadku linii telefonicznych na dużą odległość lub do rozszerzania zasięgu sieci bezprzewodowych w budynkach czy na otwartych przestrzeniach.

Chociaż repeatery są ważnym elementem w utrzymaniu niezawodności sieci i umożliwiają komunikację na długich dystansach, ich użycie wiąże się z pewnymi ograniczeniami. Z jednej strony, mogą one wzmocnić sygnał, ale jednocześnie mogą również wzmocnić szumy, co sprawia, że trudno jest rozróżnić oryginalne dane od zakłóceń. Ponadto, zbyt duża liczba repeatów w sieci może prowadzić do pogorszenia jakości sygnału, ponieważ każdy dodatkowy etap retransmisji może wprowadzać opóźnienia lub zwiększać zakłócenia.

Repeatery są kluczowe dla utrzymania niezawodności i efektywności systemów komunikacyjnych, umożliwiając przesyłanie danych na długie odległości bez utraty jakości czy integralności sygnału.

Skrypty SQL

Skrypty SQL generujące bazę danych przedstawioną na rysunku 30 pokazano poniżej. Fragment kodu tworzący bazę danych o nazwie poligrafia:

CREATE DATABASE 'poligrafia'; USE 'poligrafia';

Fragment kodu tworzący tabelę order, zawierającą atrybuty: orderid, order_name, user_id, count, status, price, itemdata, productid gdzie order id jest kluczem głównym i żaden z atrybutów, oprócz product id, nie może posiadać wartości NULL. Wartość atrybutu orderid jest unikalna i generowana automatycznie:

CREATE TABLE 'order' (

'order_id' int(11) NOT NULL auto_increment,

'order_name' varchar(255) NOT NULL default 'user_id' int(1) NOT NULL default '0′,

```
'count’ int(11) NOT NULL default '0′,

'status’ int(11) NOT NULL default '0′,

'price’ varchar(255) NOT NULL default ”,

'itemdata’ text NOT NULL,

'product_id’ int(11) default NULL,

PRIMARY KEY (’order_id’),

KEY 'user id’ (’user id’),

KEY 'product_id’ (’product_id’)

) TYPE=InnoDB ROW FORMAT=DYNAMIC AUTO INCREMENT=5 ;
```

Fragment kodu tworzący tabelę order_options, zawierającą atrybuty: option_id, order_id, atrybuty option idi orderiddomyślnie posiadają wartość NULL:

```
CREATE TABLE 'orders_options’ (

'option_id’ int(11) default NULL,

'order_id’ int(11) default NULL,

KEY 'option_id’ (’option_id’),

KEY 'order_id’ (’order_id’)

) TYPE=InnoDB ROW FORMAT=DYNAMIC;
```

Fragment kodu tworzący tabelę products, zawierającą atrybuty: id, name, price, discount, fields, description, template, sample. Kluczem głównym jest id i żaden z atrybutów nie może posiadać wartości NULL. Wartość atrybutu id jest unikalna i generowana automatycznie:

```
CREATE TABLE 'products’ (

'id’ int(11) NOT NULL auto_increment,
```

'name’ varchar(255) NOT NULL default ”,

'price’ float NOT NULL default '0′,

'discount’ text NOT NULL,

'fields’ text NOT NULL,

'description’ text NOT NULL,

'template’ varchar(255) NOT NULL default ”,

'sample’ varchar(255) NOT NULL default ”,

PRIMARY KEY (’id’),

KEY 'price’ (’price’)

) TYPE=InnoDB ROW FORMAT=DYNAMIC AUTO INCREMENT=2 ;

Fragment kodu tworzący tabelę users, zawierającą atrybuty: id, type, name, password, address, email, company. Kluczem głównym jest id i żaden z atrybutów nie może posiadać wartości NULL, Wartość atrybutu id jest unikalna i generowana automatycznie, a klucz email musi być unikalny:

CREATE TABLE 'users’ (

'id’ int(11) NOT NULL auto_increment,

'type’ varchar(32) NOT NULL default 'user’,

'name’ varchar(255) NOT NULL default ”,

'password’ varchar(255) NOT NULL default 'address’ varchar(255) NOT NULL default ’email’ varchar(255) NOT NULL default 'company’ varchar(255) NOT NULL default PRIMARY KEY (’id’),

UNIQUE KEY 'user_email_uniq’ (’email’)

) TYPE=InnoDB ROW FORMAT=DYNAMIC AUTO INCREMENT=2 ;

Fragment kodu tworzący tabelę reports, zawierającą atrybuty: id, user_id, report_data, operation_type, date_completed, date steted, odrer id. Kluczem głównym jest id i żaden z atrybutów nie może posiadać wartości NULL. Wartość atrybutu id jest unikalna i generowana automatycznie:

CREATE TABLE 'reports’ (

'id’ int(11) NOT NULL auto_increment,

'user_id’ int(11) NOT NULL,

'report_data’ text NOT NULL,

'operation_type’ int(11) NOT NULL,

'date_completed’ int(11) NOT NULL,

'date_started’ int(11) NOT NULL,

'order_id’ int(11) NOT NULL,

PRIMARY KEY (’id’),

KEY 'user_id’ (’user_id’),

KEY 'order_id’ (’order_id’)

) TYPE=InnoDB AUTO INCREMENT=12 ;

Początki i rozwój Internetu

„Większość przełomowych wynalazków jest dziełem zbiegu okoliczności, lenistwa lub efektem wojskowych programów badawczych.” [1]

Szukając genezy Internetu, można stwierdzić, iż jego powstanie nie byłoby możliwe bez wynalezienia przez Alexandra Grahama Bella telefonu. Oczywiście, historia komunikacji rozpoczęła się znacznie wcześniej – można nawet rzec, iż „od stworzenia świata”, a dokładniej od 700 roku p.n.e. Tak, już wtedy Grecy zaczęli udomawiać gołębie pocztowe i uczyli je przesyłać informacje

Internet powstał dzięki względom militarnym. Gdyby nie Zimna Wojna i wystrzelenie przez Rosjan w 1957 roku pierwszego sztucznego satelitę ziemi, Sputnika, czas powstania Internetu nastąpiłby zapewne znacznie później.

Mocno zaniepokojony tym faktem Departament Obrony USA (DoD – Departament of Defense) powołał specjalną agencję ARPA (Zaawansowanych Przedsięwzięć Badawczych), której zadaniem było zapewnienie Stanom Zjednoczonym wiodącej roli w wykorzystaniu nauki i techniki dla potrzeb militarnych. W tym samym czasie, w innej amerykańskiej instytucji zajmującej się zagadnieniami bezpieczeństwa narodowego zaczęto rozważać kwestie funkcjonowania władz i dowództwa armii USA, w wypadku wojny nuklearnej. ARPA miała na celu stworzyć sieć komunikacyjną dla celów wojskowych, która mogłaby zadziałać nawet w sytuacji zniszczenia tradycyjnych środków komunikacji. Jedynie taki system mógł zagwarantować, że w przypadku globalnego konfliktu zostanie zachowana chociażby częściowa możliwość przekazywania rozkazów pomiędzy poszczególnymi jednostkami militarnymi.

Idea ta zakładała brak centralnego punktu dowodzenia, a jej funkcjonowanie miało być analogiczne do sieci telefonicznej. W tym celu Rand Paul Baran z firmy RAND Corporation (agencja rządowa) otrzymał zlecenie od U.S. Air Force na zrealizowanie wyznaczonego celu. Efektem pracy Baran-a była dokumentacja („On Distributed Communications Networks”) opisująca kilka rozwiązań, a sieć oparta na wymianie pakietów (PSN – Packet Switching Network) W ramach tego eksperymentu Departament Obrony sfinansował badania oraz wyłożył pieniądze na powstanie ośrodków komputerowych, a w 1969 roku powstała eksperymentalna sieć ARPAnet, składająca się z czterech połączonych ze sobą komputerów mieszczących się w wybranych instytucjach naukowych: Uniwersytet Kalifornijski w Los Angeles, Uniwersytet Kalifornijski w Santa Barbara, Instytut Stanforda (SRI – Stanford Research Institute) oraz Uniwersytet Stanowy Utah.

Zgodnie z założeniami, sieć była zdecentralizowana, w architekturze typu peer-to-peer (równy z równym) – każdy z komputerów był równorzędny i połączony ze wszystkimi pozostałymi komputerami, toteż w przypadku awarii jednego z nich, mogła ona nadal funkcjonować.

Idea Paula Barana polegała na dzieleniu partii informacji na mniejsze jednostki (tzw. pakiety), następnie przesyłaniu ich osobno przez sieć oraz ponownemu łączeniu ze sobą dopiero po dotarciu do odbiorcy.

Zalety takiego rozwiązania:

- możliwość jednoczesnego korzystania z łącza przez więcej niż jednego użytkownika,
- zmniejszenie obciążenia sieci oraz redukcja błędów, gdyż w przypadku niepoprawnej transmisji wystarczy przesłać ponownie jedynie ten pakiet, który nie dotarł lub którego zawartość była przekłamana,
- polepszenie przepustowości – pakiety mogą wędrować po sieci różnymi drogami, w zależności od tego, która z nich jest w danej chwili mniej obciążona.

Pod koniec września 1969 roku odbyła się pierwsza próba zdalnego połączenia pomiędzy komputerami w Stanford i Los Angeles. Trwała ona tylko do czasu wpisania dwóch liter: “l” oraz “o” (początek słowa login). Przy przesyłaniu litery “g” jeden z komputerów się zawiesił. Sukces jednak został osiągnięty – prace teoretyczne zaczęły ocierać się o praktykę, więc projekt nie został zawieszony.

Pierwszymi usługami sieciowymi były FTP (File Transfer Protocol) i FTP, czyli kolejno zdalne logowanie i transmisja plików. W następnej kolejności pojawiła się możliwość przesyłania poczty elektronicznej (e-mail) – pierwszy program do jej obsługi został napisany przez Raya Tomlinsona (BBN Technologies), który wprowadził popularny znak “małpki” – @.

W 1972 roku w trakcie międzynarodowej konferencji poświęconej komunikacji pomiędzy komputerami (International Conference on Computer Communications) – odbyła się pierwsza publiczna demonstracja ARPAnet’u.

W sieci ARPAnet były już 23 hosty (serwery) oraz 15 węzłów, na które składały się instytucje akademickie i rządowe, a węzeł na Hawajach został podłączony przez łącze satelitarne. Do transmisji danych wykorzystywano protokół NCP – Network Control Protocol, który pozwalał na komunikację pomiędzy komputerami podłączonymi do tej samej sieci. Nazwa agencji ARPA została zmieniona na DARPA (Defense Advanced Research Projects Agency).

Rok później zaczęto coraz poważniej myśleć o połączeniach międzynarodowych. W celu ustalenia wspólnego protokołu powołano Internetwork Working Group, której przewodniczył Vinton Cerf określony później mianem ojca internetu. Cerf wraz z Bob’em Kahn’em rozpoczęli pracę nad protokołem (nazwanym później TCP/IP – Transmission Control Protocol/Internet Protocol), który pozwoli odmiennym sieciom komputerowym łączyć i porozumiewać się ze sobą.

W 1973 roku powstało pierwsze połączenie międzynarodowe z USA do Wielkiej Brytanii (London College University) przez Norwegię (Royal Radar Establishment). Tym sposobem sieć ARPAnet stała się siecią międzynarodową.

W 1974 po raz pierwszy w raporcie Cerfa i Kahn'a zostało użyte słowo Internet.

W 1975 roku powstała pierwsza lista mailingowa, a rok później królowa Elżbieta II wysyła list przy użyciu poczty elektronicznej.

W roku 1975 kierownictwo ARPA zdecydowało o zmianie statusu sieci z eksperymentalnej na użytkową [5]

Dr. Robert M. Metcalfe w roku 1976, dzięki kablowi koncentrycznemu, który umożliwiał transfer danych w niezwykle szybkim tempie, rozpoczął erę Ethernetu. Jak się później okazało, pomysł ten miał zasadnicze znaczenie w rozwoju sieci LAN.

Tego samego roku, Departament Obrony rozpoczął doświadczenie z TCP/IP i szybko padała decyzja, iż należy go wdrożyć w ARPAnet.

W roku 1979, Tom Truscott i James Ellis z Uniwersytetu Duke oraz Steve Bellovin z Uniwersytetu Północnej Karoliny sttworzyli tekstowe grupy dyskusyjne (USENET).

Popularyzacja sieci spowodowała iż zaczęło się robić gęsto i coraz bardziej różnorodnie pod względem sprzętowym, zaczęły powstawać nowe niezależne sieci. W 1980 roku już 400 serwerów było połączonych siecią ARPAnet.

W 1981 roku powstała CSNET (Computer Science Network) – sieć przeznaczona dla naukowców nie mających połączenia z ARPAnet'em oraz BITNET („Because It's Time NETwork") łącząca City University of New York z Uniwersytetem w Yale. Sieć Bitnet nie używała standardowego protokołu, ale stała się popularna dzięki usłudze Listserv – listom dyskusyjnym o dużo większych możliwościach od oferowanych przez ARPAnet.

W 1982 roku powstała w Europie sieć Eunet (European Unix Network), pozwalająca na korzystanie z usług poczty elektronicznej oraz USENET'u. Protokół TCP/IP zostajł wprowadzony jako standard dla APRAnet.

1 stycznia 1983 roku sieć ARPAnet została rozdzielona na dwie części: militarną – MILNET oraz cywilną – ARPAnet, czyli późniejszy NSFnet. Obie sieci były jednak połączone ze sobą przy pomocy bramy (gateway). Powstało również połączenie pomiędzy ARPAnet a CSNET, co uważa się za początek Internetu. Prawie jednocześnie powstały połączenia do Europy, Ameryki Południowej, Japonii i Australii. Każda maszyna dołączona do APRAnet'u musiała używać TCP/IP, który stał się protokołem podstawowym i całkowicie zastąpił NCP.

Internet został przekazany przez armię Narodowemu Funduszowi Nauki (NSF).

W tym samym roku została utworzona EARN (European Academic and Research Network) – Europejska Akademicka i Badawcza Sieć Komputerowa będąca odpowiednikiem Bitnet'u.

Listopad 1983. Na Uniwersytecie Wisconsin (University of Wisconsin) został opracowany system nazw domen DNS, który umożliwiał pakietom kierować się pod nazwę domeny, która była tłumaczona na odpowiadający jej adres IP.

W 1984, w sieci było już ponad 1000 serwerów. W Wielkiej Brytanii powstała JANET (Joint Academic Network). [3]

Tom Jennings stworzył sieć FidoNet – pierwszą na świecie, rozległą sieć komputerową, z rozproszonym zarządzaniem i dostępną dla każdego komputera. [6]

W roku 1985, NSF zaczęła rozmieszczać nowe, szybkie linie T1 o przepustowości 1,544Mbps. Plany zakładały zakończenie prac do

1988 roku. [4]

„W lipcu 1986 roku została stworzona NSFNET (National Science Foundation) – amerykańska ogólnokrajowa sieć szkieletowa o przepustowości 56 Kbps, łącząca początkowo pięć superkomputerów z ośrodków uniwersyteckich w Cornell, Illinois, Princeton, Pittsburgh i San Diego. Sieć ta rozwijała się bardzo szybko. Przyłączały się do niej także inne kraje tworzące u siebie analogiczne sieci szkieletowe. ”[1]

Pod koniec roku 1986 do sieci Internet podpiętych było już ponad 5000 komputerów, liczba grup USENET zwiększyła się do 241.[7]

Jeff Case, Mark Fedor, Martin Schoffstall i James Davin w sierpniu 1987 zaprezentowali stworzony przez nich samych protokół SGMP (Simple Gateway Monitoring Protocol), który później został zastąpiony protokołem SNMP. Protokoły te miały służyć do nadzoru i zarządzania różnymi elementami sieci komputerowych.

W tym samym czasie został opublikowany zbiór dokumentów które opisuj ą różne standardy, protokoły i procedury związane z sieciami komputerowymi i Internetem – RFC („Request for Comments”).[8]

9 grudnia, sieć BITNET zaatakował wirus (The Christmas Virus), który sparaliżował pracę serwerów poczty elektronicznej. Duża część sieci została wyłączona, aby wstrzymać rozprzestrzenianie się „robaka”.

W 1988 roku, po skończeniu prac nad tworzeniem szkieletu sieci T1, okazało się, że natężenie ruchu w sieci znacznie wzrosło. NSF po raz kolejny rozpoczęło prace nad zmodernizowaniem szkieletu sieci – unowocześnienie do łącza DS-1 (1,544 Mbps), które umożliwiało ruch ponad 75 milionom pakietów dziennie. Zostało również stworzone pierwsze światłowodowe połączenie transatlantyckie pomiędzy Północną Ameryką i Europą. Umożliwiało ono prowadzenie 40000 połączeń telefonicznych w tym samym czasie.

Stworzony przez Robert’a Morris’a Jr. pierwszy wirus internetowy „Internet Worm” zaatakował ponad 50000 komputerów podłączonych do sieci. Był to kolejny impuls przyczyniający się do zawiązania CERT (Computer Emergency Response Team) – organizacji zajmującej się zapewnieniem bezpieczeństwa w sieci.

W tym samym roku Jarkko Oikarinen, student z Uniwersytetu Oulu w Finlandii, stworzył usługę IRC (Internet Relay Chat) umożliwiającą prowadzenie rozmów w czasie rzeczywistym.

W 1989 roku powstała pierwsza specyfikacja protokołu PPP (Point to Point Protocol), która została opublikowana w raporcie RFC 1134. Do dziś niemal wszyscy użytkownicy Internetu używający dostępu telefonicznego, używają protokołu PPP.[3]

Liczba serwerów w Internecie grubo przekroczyła 100 000. Przy takiej ilości komputerów poważnym problemem stało się znalezienie żądanych informacji. Przepis na poprawę tej sytuacji odnalazł w 1990 roku Peter Deutsch. Wraz ze swoimi współpracownikami z uniwersytetu McGill w Montrealu (Alan Emtage i Bill Heelan) stworzył pierwszy katalog zasobów sieciowych. Program ARCHIE przeglądał od czasu do czasu znane serwery FTP i tworzył indeks ich zawartości z możliwością wyszukiwania plików – wkrótce po tym powstało wiele serwerów oferujących tą usługę.

Lata 90’ można nazwać przełomowym okresem dla rozwoju Internetu. ARPAnet skończył swoją działalność, zarząd nad Internetem przejął NSFnet. Liczba serwerów przekroczyła 300 000, a grup dyskusyjnych było już około 1 000.

CERN (“Centre European pour la Recherche Nucleaire”) –

szwajcarski instytut z siedzibą w Genewie udostępnił wyniki swoich badań naukowcom z całego świata. Tim Berners-Lee korzystając z udostępnionych informacji stworzył pierwszą przeglądarkę tekstową do WWW (Word Wide Web). Wpadł on na pomysł, aby powiązać ze sobą dokumenty znajdujące się na serwerach WWW.

Szwajcarski instytut CERN ("Centre European pour la Recherche Nucleaire" później "European Laboratory for Particle Physics") w Genewie poczuł nieodpartą potrzebę udostępnienia wyników swoich badań naukowcom z całego świata. Tim Berners-Lee wpadł na pomysł powiązania ze sobą dokumentów znajduj ących się na serwerach WWW (World Wide Web) przy pomocy łączy hipertekstowych, co umożliwiło połączenie tekstu, grafiki oraz dźwięku. W 1991 roku stworzył on pierwszą przeglądarkę tekstową do WWW. Pierwszy amerykański serwer WWW powstał w Stanford Linear Accelerator Center w Kalifornii.

1991 r. – NSFnet zniósł zakaz używania Internetu do celów komercyjnych oraz, co dla nas najważniejsze, Polska zostaje wreszcie przyłączona do Internetu.

Pojawiły się systemy WAIS (Wide Area Information Server), czyli systemy rozległych baz danych stworzone przez Brewstera Kahle. WAIS indeksowały pełną zawartość różnych baz danych, dokumentów RFC oraz plików FAQ (Frequently Asked Questions – Najczęściej Zadawane Pytania) list dyskusyjnych.

Na Uniwersytecie Minnesota w USA powstał Gopher, czyli system informacyjny udostępniający różne zasoby (n.p.: pliki tekstowe, binarne, graficzne) oraz usługi sieciowe. Gopher bardzo szybko zyskał dużą popularność, gdyż był on dużo prostszy w obsłudze od Archie'go czy też systemu WAIS. Został on zaadaptowany przez większość ośrodków komputerowych na świecie. Obecnie rolę Gopher'a w całości przejęła usługa WWW. Powstał standard PGP (Pretty Good Privacy) umożliwiający szyfrowanie przesyłek (Philip Zimmerman).

W 1992 roku kolejne ośrodki naukowe tworzyły swoje serwery WWW, pod koniec roku było ich już 50.

Liczba hostów w sieci przekroczyła milion. Powstała Społeczność Internetowa (Internet Society) – ISOC: www.isoc.org/, która obecnie skupia 150 organizacji i 6 000 indywidualnych członków z ponad 100 krajów.

W 1993 roku Marc Andreessen wraz z zespołem NCSA (National Center For Supercomputing Applications) stworzyli Mosaic – pierwszą przeglądarkę graficzną do odczytywania stron WWW. W sieci pojawiła się strona internetowa Białego Domu. Rozpoczęła się wielka kariera stron internetowych – serwerów WWW było już pięć razy więcej niż rok wcześniej.

Pierwsza międzynarodowa konferencja poświęcona WWW („Woodstock of the Web"), odbyła się w 1994 roku w instytucie CERN i zainteresowała ponad 600 potencjalnych uczestników, jednakże tylko 400 osób mogło wziąć w niej udział.

Od tego samego roku można przez Internet słuchać radia oraz zamówić pizzę z Pizza Hut, w sieci pojawia się także pierwszy bank.

W październiku 1994 roku z inicjatywy Tima Berners-Lee w Massachusetts Institute of Technology powstała organizacja World Wide Web Consortium (W3C – www.w3c.org). Udział w stworzeniu tej organizacji miał CERN, a wsparcia udzielili DARPA oraz Komisja Europejska. W kwietniu 1995 do organizacji dołączył INRIA (Institut National de Recherche en Informatique et Automatique) – powstał pierwszy europejski serwer W3C. W3C zajmuje się rozwojem sieci, tworzeniem nowych standardów i technologii oraz zatwierdzaniem oficjalnych specyfikacji (np. języka HTML, arkuszy stylów).

Członkami W3C są naukowcy, programiści, twórcy stron internetowych, firmy, instytucje oraz stowarzyszenia (n.p: HTML Writers Guild). Działanie W3C jest finansowane przez większość znaczących korporacji zajmujących się tworzeniem sprzętu

i oprogramowania komputerowego (m.i.n: Microsoft, Intel, Netscape, Apple) oraz inne firmy żywotnie zainteresowane rozwojem Internetu (np.: Boening, Canal+).

W latach 90-tych pojawiły się nowe technologie: Java, JavaScript, Internet Phone, ActiveX, VRML (Virtual Environments), RealAudio (przesyłanie dźwięku), WebTV, ASP oraz popularny stał się dostęp do sieci przez modem.

W 1995 roku NSFnet przekształciła się w sieć badawczą, Internet w komercyjną. Powstały przeglądarki Netscape Navigator oraz Internet Explorer. Pojawiły się firmy Compuserve, America Online, Prodigy zajmujące się oferowaniem dostępu do Internetu. Do sieci wkroczyła komercja i Watykan, pojawiały się pierwsze sklepy internetowe.

Rozwój Internetu został uzależniony od wsparcia go przez rządy poszczególnych krajów i przedsięwzięcia o charakterze międzynarodowym. W związku z tym powstały dwa nowe projekty, przygotowane w USA i Europie. Pierwszy z nich to inicjatywa Narodowej Infrastruktury Informacyjnej opracowana przez rząd federalny Stanów Zjednoczonych, a drugi – to projekt parlamentu europejskiego, znany pod nazwą Shopping 2000. Został podjęty w roku 1996, w ramach programu Eureka. Jego zadaniem było rozwiązanie podstawowych problemów związanych z handlem elektronicznym, przede wszystkim zapewnić bezpieczeństwo firmom uczestniczącym w takim handlu. W zakres przedsięwzięcia wchodzi też stworzenie odpowiedniej infrastruktury teleinformatycznej. W projekcie uczestniczyło 15 firm i organizacji, w tym firmy informatyczne.[5]

W 1996 roku powstały wyszukiwarki Lycos i Yahoo. Do Sieci przyłączonych było ponad 9 milionów hostów. Larry Page i Sergey Brin rozpoczęli pracę nad wyszukiwarką stron internetowych, nazwaną BackRub. Było to unikalne rozwiązanie do analizy linków, które prowadziły do określonej strony. W późniejszym czasie nazwa wyszukiwarki została zmieniona na „Google”. Network Solutions usunęło ze swoich serwerów DNS ponad 9270 wpisów z powodu nieopłaconych nazw domen. [3]

W 1997 liczba komputerów podłączonych do Internetu przekroczyła 19,5 miliona, istniało ponad 1,3 miliona stron WWW oraz ponad 71 tysięcy grup dyskusyjnych. Pracownik firmy Network Solutions popełnił poważny błąd, uszkadzając bazę danych DNS, w skutek czego nieosiągalne stały się domeny z końcówkami .com oraz .net.

W roku 1998 US Commerce Departament opracował „Green Paper” – dokumentację wyjaśniającą jak powinien funkcjonować system rejestracji domen. Firma Netscape Communications zapowiedziała, iż ma w planach nieodpłatne udostępnienie kodu źródłowego swojej przeglądarki stron WWW (Netscape Communicator) w Internecie. International Telecommunication Union oświadczyło, iż ustalili warunki zaimplementowania protokołu V.90 w modemach. Pod koniec roku została zarejestrowana trzymilionowa domena. [3]

W 1999 roku First Internet Bank of Indiana, zaoferował całodobową pełną obsługę przez Internet. Był to pierwszy bank dostępny tylko przez Internet. W marcu 1999 roku makro-wirus Melissa szybko rozprzestrzenił się w formie wiadomości e-mail z dołączonym dokumentem programu Word, zawierającym wirus makra. „Wirus był inicjowany po otwarciu załącznika. Następnie mógł on spowodować wysłanie dokumentu (razem z samym wirusem) w wiadomości e-mail do pierwszych 50 osób z książki adresowej użytkownika. Wiadomość e-mail zawierała miły tekst z nazwiskiem użytkownika, odbiorca otwierał więc dokument, sądząc, że był nieszkodliwy. Wtedy wirus powodował utworzenie 50 nowych wiadomości do osób z książki adresowej odbiorcy. W rezultacie wirus Melissa był najszybciej jak dotąd rozprzestrzeniającym się wirusem. Zmusił on wiele firm do zamknięcia systemów poczty e- mail.”[9] Ilość zarejestrowanych domen internetowych przekroczyła pięć milionów.

W 2000 powstała technologia WAP (Wireless Application Protocol), która miała przyczynić się do usprawnienia przesyłania danych w sieci telefonii bezprzewodowej. Po raz pierwszy zaprezentowano tą technologię już rok wcześniej. Możliwa stała się rejestracja domen w językach chińskim, japońskim i koreańskim. Liczba hostów w sieci przekroczyła 90 milionów.

---

[1] „Poradnik Webdesignera", Agnieszka Richter, Kalia Studio, kaila.biz/design/htm/article/historia.htm

[3] "History of the Internet and Web (from 700 BC)", Anthony Anderberg, http://www.anderbergfamily.net/ant/history/

[4] "The History of the Internet", Dave Kristula, http://www.davesite.com/webstation/net- shtml

[5] „Marketing wirtualny", Andrzej Sznajder, Kraków: Oficyna Ekonomiczna 2000.

[6] „Wikipedia PL", http://pl.wikipedia.org/

[7] "Hobbes' Internet Timeline1", Robert Hobbes' Zakon, Zakon Group LLC, http://www.zakon.org/robert/internet/timeline/

[8] Algorytmy.pl, http://algorytmy.pl/encyklopedia/

RIP – Routing Information Protocol

Protokół RIP jest protokołem routingu, w którym zastosowano algorytm distance-vector. Jest on szeroko stosowany w sieciach jako protokół wewnętrzny IGP (Interior Gateway Protocol), co oznacza, że wykonuje routing pojedynczym autonomicznym systemem albo protokołem zewnętrznym EGP (Exterior Gateway Protocol) – wykonuje routing pomiędzy różnymi autonomicznymi systemami. Protokół RIP jest obecnie szeroko wykorzystywany w Internecie.

Protokół RIP (Routing-Information Protocol) jest używany w sieciach jako podstawowa metoda wymiany informacji o routingu pomiędzy routerami. Specyfikacje protokołu RIP definiują dwa dokumenty RFC (Request For Comments) 1058 i 1723. RFC 1058 opisuje pierwszą implementację protokołu, natomiast jego wersję zaktualizowaną opisuje dokument RFC 1723. Opierając się na protokole RIP routery podejmują następujące działania:

1. Żądają aktualnych informacji o routingu od innych routerów i na ich podstawie aktualizują tablice routingu.
2. Odpowiadają na podobne żądanie innych routerów.
3. W ściśle określonych przedziałach czasu rozsyłają informacje o swojej obecności, informując inne routery o aktualnej konfiguracji połączeń międzysieciowych.
4. W przypadku wykrycia zmian w konfiguracji sieci rozsyłają stosowną informację.

Algorytm distance-vector

Distance-vector optymalizuje wybór trasy przy kryterium najmniejszej liczby skoków (hops) niezbędnych do osiągnięcia miejsca przeznaczenia (destination) lub kosztu ścieżki prowadzącej do miejsca przeznaczenia.

Format pakietu RIP

Polecenie (*Command*) - wskazuje, czy pakiet jest zgłoszeniem (*request*), czy odpowiedzią (*response*). Zgłoszenie pyta, czy router wysłał całą tablicę routingu czy jej część.
Odpowiedzią może być dobrowolne periodyczne uaktualnienie routingu lub odpowiedź na zgłoszenie. Przy przenoszeniu obszernych tablic routingu zachodzi potrzeba użycia wielu pakietów.
Numer wersji (*Version number*) - określa użytą wersję protokołu RIP. Pole to może sygnalizować różne potencjalnie niezgodne wersje.
Rezerwa (*Unused*) - pole nie używane, przyjmuje wartość zero.
Identyfikator AFI (*Address-Family Identifier*) - określa użyty adres rodziny.
Protokół RIP jest przeznaczony do przenoszenia informacji o routingu dla wielu różnych protokołów.
Każde wejście ma identyfikator AFI wskazujący typ specyfikowanego adresu. Dla protokołu IP identyfikator API przyjmuje wartość 2.
Adres (*Address*) - określa adres IP dla wejścia.
Miara (*Metric*) - wskazuje liczbę przejść pomiędzy sieciami (routerami), które pojawiły się na drodze do miejsca przeznaczenie. Wartość miary mieści się w przedziale od 1 do 15. Dla trasy prowadzącej donikąd przyjmuje wartość 16.
Uwaga: W jednym pakiecie RIP może pojawić się nie więcej niż 25 identyfikatorów API, adresów i pól zawierających miary.

Uaktualnianie routingu

Protokół RIP wysyła komunikaty uaktualniające w określonych, stałych przedziałach czasu, na przykład co 30 s, lub w przypadku pojawienia się zmian w topologii sieci. Router po przyjęciu uaktualnienia routingu, które dotyczy zmian jednego z wejść, uaktualnia tablicę routingu (routing table), by odzwierciedlić nową trasę. Wartość miary przypisanej danej ścieżce wzrasta o jeden i jako następny skok jest wskazany nadawca. Routery RIP utrzymują do miejsca przeznaczenia tylko najlepszą trasę, to jest trasę z najmniejszą liczbą skoków. Router niezwłocznie po uaktualnieniu swojej tablicy routingu wysyła informacje o zmianie do pozostałych routerów w sieci. Są one wysyłane niezależnie od regularnie wysyłanych uaktualnień.

Miara routingu protokołu RIP

Jedyną miarą używaną przez protokół RIP do mierzenia odległości pomiędzy źródłem a miejscem przeznaczenia jest zliczanie skoków (hop-count). Każdemu skokowi na drodze od źródła do miejsca przeznaczenia zostaje przypisana wartość, najczęściej 1. Gdy router przyjmie uaktualnienie tablicy routingu, które zawiera nowe lub zmienione wejście sieciowego miejsca przeznaczenia, to dodaje jedynkę do wartości miary wskazanej w uaktualnieniu i wpisuje zmianę do tablicy routingu. Adresem następnego skoku jest adres IP nadawcy. Protokół RIP, dzięki ograniczeniu liczby skoków, które mogą pojawić się pomiędzy źródłem a miejscem przeznaczenia, zapobiega przesyłaniu strumienia danych bez końca w pętli. Maksymalna liczba skoków na ścieżce wynosi 15. Jeśli router przyjmie uaktualnienie routingu, które zawiera nowe lub zmienione wejście, i jeśli po zwiększeniu miary o jeden nastąpi przekroczenie granicy 15 skoków, to takie miejsce przeznaczenia w sieci staje się nieosiągalne.

Format pakietu RIP-2

W pakiecie RIP-2 (RFC 1723) można zawrzeć więcej informacji niż w pakiecie RIP. Specyfikacja pakietu RIP-2 zawiera proste mechanizmy autoryzacji.

Polecenie (*Command*) - wskazuje, czy pakiet jest zgłoszeniem (request) czy odpowiedzią (response). Zgłoszenie pyta, czy router wysłał całą tablicę routingu czy jej część. Odpowiedzią może być dobrowolne periodyczne uaktualnienie routingu lub odpowiedź na zgłoszenie. Przy przenoszeniu obszernych tablic routingu zachodzi potrzeba użycia wielu pakietów.

Wersja (*Version*) - określa użytą wersję protokołu RIP. W pakiecie RIP implementującym jakiekolwiek pola RIP-2 lub mechanizm autoryzacji jest ustawiona wartość 2.

Rezerwa (*Unused*) - pole nie używane, przyjmuje wartość zero.

Identyfikator AFI (*Address-Family Identifier*) - określa użyty adres rodziny. Protokół RIP jest przeznaczony do przenoszenia informacji o routingu dla wielu różnych protokołów. Każde wejście ma identyfikator AFI wskazujący typ specyfikowanego adresu.

Dla protokołu IP identyfikator API przyjmuje wartość 2. Jeśli identyfikator AFI dla pierwszego wejścia w komunikacie przyjmuje wartość 0xFFFF, to oznacza, że wejście zawiera informację o autoryzacji, czyli typ autoryzacji i hasło.

Znacznik trasy (*Route Tag*) - pozwala rozróżnić trasę wewnętrzną (poznawaną przez protokół RIP) od zewnętrznej (poznawanej przez inne protokoły).

Adres (*Address*) - określa adres IP dla wejścia.

Maska podsieci (*Subnet Mask*) - zawiera maskę podsieci dla określonego wejścia. Jeśli pole to przyjmuje wartość 0, to oznacza niewyspecyfikowanie maski podsieci.

Następny skok (*Next Hop*) - Wskazuje adres IP następnego skoku (routera), do którego pakiet dla danego wejścia powinien być skierowany.

Miara (*Metric*) - wskazuje liczbę przejść pomiędzy sieciami (routerami), które pojawiły się na drodze do miejsca przeznaczenie. Wartość miary mieści się w przedziale od 1 do 15. Dla trasy prowadzącej donikąd przyjmuje wartość 16.

Uwaga: W jednym pakiecie RIP możne pojawić się nie więcej niż 25 identyfikatorów API, adresów i pól zawierających miary.

Stabilność protokołu RIP

W celu dostosowania się do szybkich zmian topologii sieci protokół RIP wyposażono, podobnie jak i inne protokoły routingujące, w mechanizmy stabilizujące. Na przykład, by zapobiec skutkom błędnej informacji o routingu, w protokole RIP zaimplementowano mechanizmy split-horizon i hold-down. Powstaniu pętli zapobiega ograniczenie – na trasie pomiędzy źródłem a miejscem przeznaczenia – liczby skoków (do 15).

Czasomierze protokołu RIP

W celu dostosowania do potrzeb wydajności routingu, protokół RIP wyposażono w kilka czasomierzy (timers). Wśród nich są: czasomierz uaktualnienia routingu (routing-update timer), limitu czasu trasy (route timeout timer) i czyszczenia trasy (route-flush timer). Czasomierz uaktualnienia routingu wyznacza przedziały czasu pomiędzy kolejnymi okresami uaktualniania. Jest to stały przedział nie przekraczający 30 s. Do każdego wejścia do tablicy routingu jest przypisany czasomierz limitu czasu trasy; w przypadku jego wyczerpania trasa zostaje oznaczona jako nieważna. Mimo tego jest nadal utrzymywana w tablicy routingu aż do momentu, gdy zostanie wyczerpany czasomierz czyszczenia trasy.

Krosownica i koncentrator

praca dyplomowa pod tytułem Sieć komputerowa

Krosownica oraz koncentrator to dwa fundamentalne urządzenia sieciowe, które odgrywają istotną rolę w zarządzaniu i organizacji infrastruktury sieciowej, zwłaszcza w sieciach lokalnych (LAN). Choć różnią się pod względem funkcji i zastosowania, oba te elementy pełnią ważne role w przekazywaniu i organizacji sygnałów oraz danych, zapewniając poprawne działanie sieci.

Krosownica (Patch Panel)

Krosownica, znana także jako patch panel, to urządzenie służące do organizacji i zarządzania kablami sieciowymi. Jest to pasywny element infrastruktury sieciowej, który nie przetwarza sygnałów, lecz jedynie umożliwia ich fizyczne przekierowanie. Krosownice są najczęściej używane w serwerowniach oraz pomieszczeniach telekomunikacyjnych, gdzie liczba kabli sieciowych jest bardzo duża, a ich właściwe uporządkowanie jest kluczowe dla sprawnego zarządzania i konserwacji sieci. Krosownice umożliwiają łatwe łączenie i rozłączanie różnych segmentów sieci, co przyczynia się do poprawy elastyczności infrastruktury. Dzięki odpowiedniemu oznakowaniu portów, administratorzy sieci mogą łatwo zidentyfikować poszczególne połączenia, co ułatwia diagnostykę i rozwiązywanie problemów. Krosownica jest więc swoistym centralnym punktem, gdzie kończą się kable przychodzące z różnych części budynku lub obszaru i gdzie można je połączyć z odpowiednimi urządzeniami sieciowymi, takimi jak routery, przełączniki czy serwery.

Warto podkreślić, że krosownica działa na poziomie fizycznym, a więc nie zarządza ani nie analizuje danych przepływających przez kable. Jej funkcją jest jedynie porządkowanie kabli i umożliwienie prostych zmian w układzie połączeń, co jest kluczowe w kontekście rozbudowy infrastruktury oraz szybkiej reakcji na zmieniające się potrzeby sieciowe. W zależności od potrzeb, krosownice mogą obsługiwać różne rodzaje kabli, w tym przewody miedziane, takie jak kable Ethernet, oraz światłowody. Współczesne krosownice często wyposażone są w panel czołowy z gniazdami RJ45, do których można podłączać patch cordy, co umożliwia szybkie zmiany w konfiguracji sieci bez konieczności przełączania kabli w trudno dostępnych miejscach.

Składa się z rzędów punktów zakończeniowych dla stacji roboczych. Administrator sieci może w łatwy sposób łączyć, przesuwać, testować i rozłączać elementy sieci (np. stacje robocze) – poprzez zmianę połączeń w krosownicy.

Koncentrator (Concentrator Device)

Koncentrator, znany również jako hub, jest kolejnym urządzeniem sieciowym, które odgrywa istotną rolę w komunikacji w sieciach lokalnych. W odróżnieniu od krosownicy, koncentrator jest urządzeniem aktywnym, co oznacza, że przetwarza sygnały, które przez niego przepływają. Jego podstawową funkcją jest umożliwienie komunikacji między wieloma urządzeniami w sieci LAN poprzez rozdzielanie sygnału przychodzącego na wiele innych portów. Koncentrator działa na zasadzie retransmisji – kiedy otrzymuje sygnał od jednego urządzenia, przesyła go do wszystkich innych urządzeń podłączonych do koncentratora. To sprawia, że koncentrator działa w trybie półdupleksowym, co oznacza, że dane mogą być przesyłane w obu kierunkach, ale nie jednocześnie. Taka architektura może prowadzić do powstawania kolizji danych, zwłaszcza w większych sieciach, gdzie wiele urządzeń jednocześnie próbuje komunikować się ze sobą.

Ze względu na swoją prostotę, koncentratory były powszechnie używane w sieciach LAN w latach 90. XX wieku, jednak z biegiem czasu zostały stopniowo zastąpione przez bardziej zaawansowane urządzenia, takie jak przełączniki sieciowe (ang. switches), które oferują lepszą wydajność oraz zarządzanie ruchem sieciowym. W porównaniu do koncentratorów, przełączniki potrafią inteligentnie przekazywać dane tylko do konkretnego urządzenia, co zmniejsza ryzyko kolizji i poprawia ogólną wydajność sieci.

Koncentratory są nadal stosowane w niektórych małych sieciach, gdzie liczba urządzeń jest ograniczona, a wymagania dotyczące przepustowości nie są wysokie. Ich główną zaletą jest prostota oraz niska cena, co czyni je dobrym rozwiązaniem w mniej skomplikowanych środowiskach sieciowych. Jednakże, w przypadku bardziej zaawansowanych sieci, gdzie kluczowe znaczenie mają szybkość i niezawodność, koncentratory często okazują się niewystarczające, a ich ograniczenia stają się szczególnie widoczne w kontekście rosnących potrzeb na większe przepustowości oraz lepsze zarządzanie ruchem sieciowym.

Pod względem działania, koncentrator pracuje na drugiej warstwie modelu OSI – warstwie łącza danych. Nie potrafi analizować pakietów ani zarządzać ruchem sieciowym w taki sposób, jak bardziej zaawansowane urządzenia, takie jak przełączniki czy routery. Jego zadaniem jest jedynie rozdzielanie sygnałów na wszystkie porty, do których są podłączone urządzenia. W wyniku tego, każde urządzenie podłączone do koncentratora odbiera wszystkie dane wysyłane przez inne urządzenia, co może prowadzić do przeciążenia sieci i spadku wydajności w przypadku dużego ruchu.

Koncentrator jest urządzeniem służącym za centralny punkt przyłączenia terminali, komputerów lub urządzeń komunikujących. Może to być centralny punkt, w którym zbiegają się kable. Koncentrator łączy określoną liczbę linii wejściowych z pewną liczbą linii wyjściowych albo udostępnia jedno centralne połączenie komunikacyjne większej liczbie urządzeń. Koncentratory mogą być łączone ze sobą w struktury hierarchiczne. urządzenia, które są koncentratorami:

a) procesory czołowe (front-end) – jest to komputer realizujący funkcje koncentratora, zazwyczaj z większą szybkością i obsługujący większą liczbę dołączonych urządzeń;

b) huby (hubs) – koncentratory w sieciach lokalnych (opisane dalej);

c) jednostki wspólnego dostępu do portu i selektory (port sharing units) – umożliwiają większej liczbie odległych terminali korzystanie ze wspólnego połączenia modemowego z komputerem lub systemem host. Jednostka taka działa pomiędzy terminalami a modemem;

d) multipleksery – urządzenia, które przesyłają po jednej linii dane napływające z wielu innych urządzeń. Istnieje wiele typów multiplekserów, np.: multipleksery z podziałem czasu (przydziela kolejnym urządzeniom odcinki czasu w strumieniu danych), multipleksery z podziałem częstotliwości (wydzielają dla każdego urządzenia osobny kanał częstotliwości).

Krosownica i koncentrator są kluczowymi elementami infrastruktury sieciowej, które pełnią różne funkcje, ale są równie ważne dla prawidłowego funkcjonowania sieci. Krosownica, jako pasywny element, służy do porządkowania i zarządzania kablami sieciowymi, co ułatwia organizację oraz modyfikację sieci. Koncentrator z kolei umożliwia komunikację między urządzeniami w sieci, choć ze względu na swoje ograniczenia jest coraz częściej zastępowany przez bardziej zaawansowane urządzenia, takie jak przełączniki. Mimo to, zarówno krosownica, jak i koncentrator odgrywają istotną rolę w budowie i utrzymaniu efektywnych sieci lokalnych, a ich właściwe zastosowanie może znacząco wpłynąć na wydajność i niezawodność całej infrastruktury sieciowej.

Rutowanie pakietów

Rutowanie statyczne a rutowanie dynamiczne.

Zastosowanie rutingu statycznego niesie ze sobą wiele korzyści. Przede wszystkim jest przewidywalne, ponieważ wcześniej administrator sam ustala tablicę rutowania i wie dokładnie, jaką drogę przebędzie pakiet, aby osiągnąć miejsce docelowe. W przypadku zastosowania rutingu dynamicznego nie jest już takie proste określenie trasy pakietu, ponieważ protokół sam podejmuje taką decyzję na podstawie otrzymanych danych o stanie łączy oraz odległości od innych routerów. Jak już napisałem wyżej protokoły rutingu dynamicznego rozsyłają, co pewien czas informacje, na podstawie, których podejmują odpowiednie decyzje którędy przesłać pakiet, aby dotarł najszybciej do miejsca docelowego. Informacje te w pewnych okolicznościach mogą w znacznym stopniu zmonopolizować dostępne pasmo w danej sieci. Zależy to w dużym stopniu od ilości ruterów i zastosowanego protokołu rutowania dynamicznego.

Na przykład w sieci składającej się z 200 segmentów, co 30 sekund, zgodnie ze specyfikacją protokołu RIP, rutery powinny wysyłać informacje aktualizacyjne zawierające opis dostępności wszystkich 200 segmentów sieci. Jak widać rutery w ciągu 30 sekund muszą wysłać przez każdy ze swoich interfejsów informację, która zawiera około 3 Kb danych. Mnożąc to przez 200 wychodzi około 600 Kb informacji wysyłanych przez rutery w ciągu 30 sekund – przy wolnych łączach modemowych może to znacznie obciążyć pasmo w sieci.

Poza tym ruting statyczny jest prosty do skonfigurowania w małych sieciach, gdzie administrator musi jedynie powiadomić rutery o każdym z segmentów, do których dany ruter nie jest bezpośrednio podłączony.

Podstawową wadą rutingu statycznego jest to, że jest on praktycznie nie do zastosowania w dużych skomplikowanych sieciach składających się z dużej ilości ruterów i różnych rodzajów łącz. W takim przypadku niezastąpiony jest protokół rutingu dynamicznego. Głównymi zaletami rutowania dynamicznego w stosunku do rutowania statycznego są skalowalność i zdolność dopasowywania się do zmieniających się połączeń sieci.

Rutery wykorzystujące ten protokół są w stanie „uczyć się” topologii danej sieci po przez wymianę informacji o stanie łącz i podłączonych segmentach do innych ruterów.

W takim przypadku sieć może sama reagować na zachodzące w niej uszkodzenia i na podstawie informacji zawartych w uaktualnieniach rozwiązywać te problemy.

Oczywiście protokół dynamicznego rutingu ma tez swoje wady, największą wadą jest zawiłość działania sieci, aby ruter był w stanie określić najlepszą
i najkrótszą trasę do docelowego segmentu sieci, musi przetworzyć dużo informacji nadchodzących od innych rutenów. Ponadto, aby ruter reagował
na zmieniające się warunki w sieci musi mieć możliwość usuwania starych
i bezużytecznych informacji o trasach ze swojej tablicy rutingu. Sposób, w jaki będzie to robił jeszcze bardziej komplikuje budowę takiego protokółu.

Stopień komplikacji protokółu prowadzi do błędów w jego poprawnej implementacji lub różnić w interpretacji tego protokółu w urządzeniach różnych producentów.

Klasyfikacja dynamicznych protokołów rutowania.

Protokoły rutingu dynamicznego można klasyfikować na kilka sposobów:

- protokoły zewnętrzne w porównaniu z protokołami wewnętrznymi;
- protokoły typu dystans-wektor w porównaniu z protokółami stanu łącza.

Pierwsza klasyfikacja opiera się na tym, w jakiej części sieci je stosujemy, druga opiera się na rodzaju informacji, jakie protokół wymienia oraz sposobie, w jaki każdy z ruterów podejmuje decyzję o wprowadzeniu do swojej tablicy rutowania otrzymanych informacji.

Protokoły zewnętrzne a wewnętrzne.

Dynamiczne protokoły rutowania są klasyfikowane jako, Exterior Gateway *Protocol (*EGP) lub *Interior Gateway Protocol* (IGP).

Zewnętrzny protokół odpowiada za wymianę informacji o rutowaniu pomiędzy dwiema niezależnymi sieciami, takimi jak sieci dwóch korporacji. Każda z tych jednostek ma niezależną infrastrukturę sieciową i wykorzystuje EGP
do przesyłania informacji o rutowaniu do innych podobnych jednostek. Najpopularniejszym obecnie zewnętrznym protokołem jest *Border Gateway Protocol* (BGP). Jest on podstawowym protokołem stosowanym pomiędzy sieciami tworzącymi globalna sieć Internet i specjalnie w tym celu został opracowany.

W przeciwieństwie do protokołu opisanego powyżej IGP jest stosowany wewnątrz sieci lub pomiędzy blisko współpracującymi sieciami. Protokół ten został tak stworzony, aby jego prosta budowa jak w najmniejszym stopniu obciążała rutery. Główną wadą tego typu protokołów jest to ze nie są one
w stanie obsługiwać rozrastających się sieci. Najczęściej stosowanymi w sieciach IP protokołami są:

- *Ruting Information Protocol (RIP),*
- *Open Shortest Path First (OSPF),*
- *Enhanced Interior Gateway Routing Protocol*

Pierwsze dwa protokoły są otwartymi standardami, które zostały użyte lub wymyślone przez społeczność sieci Internet a trzeci jest protokołem firmowym, opracowanym przez firmę CISCO Systems i stosowane w ruterach tej firmy.

Protokoły dystans – wektor a protokoły stanu łącza.

Innym sposobem klasyfikowania dynamicznych protokołów rutowania jest opieranie się na informacjach jakie przekazują pomiędzy sobą rutery oraz
na sposobie w jaki wykorzystują one informacje znajdujące się w ich tablicach rutowania. Większość protokołów należy do jednej z wymienionych kategorii.
W protokołach dystans-wektor rutery regularnie wysyła do sąsiadów dwie części informacji, które posiada na temat adresów przeznaczenia do których zna drogę.
Pierwsza część informacji mówi sąsiadom rutera jak daleko jest adres przeznaczenia, a druga informuje o tym w jakim kierunku (wektorze) należy kierować pakiety aby dotarły do punktu przeznaczenia.[1] Ruter kolejnego przeskoku wskazuje kierunek , który należy wykorzystać, aby pakiety osiągnęły punkt przeznaczenia, a wymieniona informacja zwykle przyjmuje formę: *„wyślij to do mnie, bo ja wiem , jak to przesłać dalej”.*

Na przykład: uaktualnienia tras RIP zawierają po prostu listę adresów do których rozgłaszający je ruter zna trasę, a także odległość, w jakiej te adresy się znajdują.

Na podstawie odbieranych uaktualnień inny ruter wnioskuje że adresem kolejnego przeskoku prowadzącego do danego miejsca w sieci jest rozgłaszający informacje ruter. Uaktualnienie może jednak przyjąć formę przekazu typu: *„prześlij to do innego rutera, który wie, jak się tam dostać.”*

Ta druga forma uaktualnienia jest zwykle wykorzystywana wtedy, kiedy ruter przez który można dotrzeć do danego miejsca, nie może lub nie będzie mógł rozgłaszać informacji z powodu awarii sieci. Nie wszystkie jednak protokoły rutowania obsługują ten typ uaktualnienia.

Druga część protokołu, którą jest informacja o odległości, stanowi o różnicy między protokołem dystans-wektor a innymi protokołami. W każdym z przypadków protokół używa pewnej *miary,* aby poinformować odbierające informacje rutery o tym, jak daleko jest adres przeznaczenia. Miara ta może być prawdziwym wskaźnikiem określającym odległość (na przykład okresowe sprawdzanie czasu podróży pakietu do miejsca przeznaczenia), czymś, co w przybliżeniu określa odległość (tak jak liczba przeskoków), lub może to być inna wartość nie związana wcale z odległością. Zamiast tego można na przykład mierzyć koszty danej drogi do miejsca przeznaczenia. Określanie tej wartości może być również wykonywane na podstawie skomplikowanych obliczeń, w których brane są pod uwagę czynniki takie jak obciążenie sieci, pasmo łącza, opóźnienie łącza i inne wartości opisujące ruter. Wartość ta może również zawierać wagę, określaną przez administratora sieci w celu wskazania jednej z tras jako preferowanej w stosunku do innych.

W każdym z przypadków wartość miary kosztu pozwala ruterowi wybrać najlepszą trasę ze wszystkich informacji, jakie do niego docierają w postaci rozgłaszanych informacji o trasach. Wybór dokonywany jest na podstawie porównania odległości podanej w różnych rozgłaszanych trasach. Sposób, w jaki dokonywane jest to porównanie, zależy od tego, jak liczona lub określana jest wartość przekazywanej miary. Na przykład miary w trasach przekazywanych w uaktualnieniach RIP są określone jako liczba przeskoków, gdzie jeden przeskok oznacza obsługę pakietu przez jeden ruter na drodze do miejsca przeznaczenia. Miejsce przeznaczenia z podaną liczbą przeskoków równą 16 uznaje się za nieosiągalne. Kiedy jakiś ruter odbiera uaktualnienia RIP od różnych ruterów, odnoszące się do tej samej sieci, wybiera trasę, która ma najmniejszą miarę. Jeśli miara ta jest mniejsza od tej, którą przechowuje w swojej tablicy rutowania, ruter wymienia trasę do danej sieci na nową, zakładając, że uzyskane z innego rutera informacje są aktualne.

Aby informacja o trasach do różnych podsieci mogła być propagowana poprzez sieć każdy ruter umieszcza w rozgłaszanych komunikatach wszystkie kierunki, do których jest bezpośrednio dołączony, a także trasy do miejsc przeznaczenia, o których dowiedział się od innych ruterów. Kiedy ruter zaczyna przekazywać dalej informacje o trasach, o których dowiedział się od innych ruterów, to konieczny jest algorytm wchodzący w skład protokołu rutowania, który dokona odpowiedniego zwiększenia miary dla danej trasy. W przypadku protokołu RIP oznacza to, że zanim ruter rozpowszechni informację, którą wcześniej uzyskał z innego rutera, do metryki każdej z tych informacji dodaje jeden przeskok. Dzięki takiemu algorytmowi miara rośnie, gdy zwiększa się odległość od miejsca przeznaczenia wskazywanego przez zapis w tablicy rutowania.

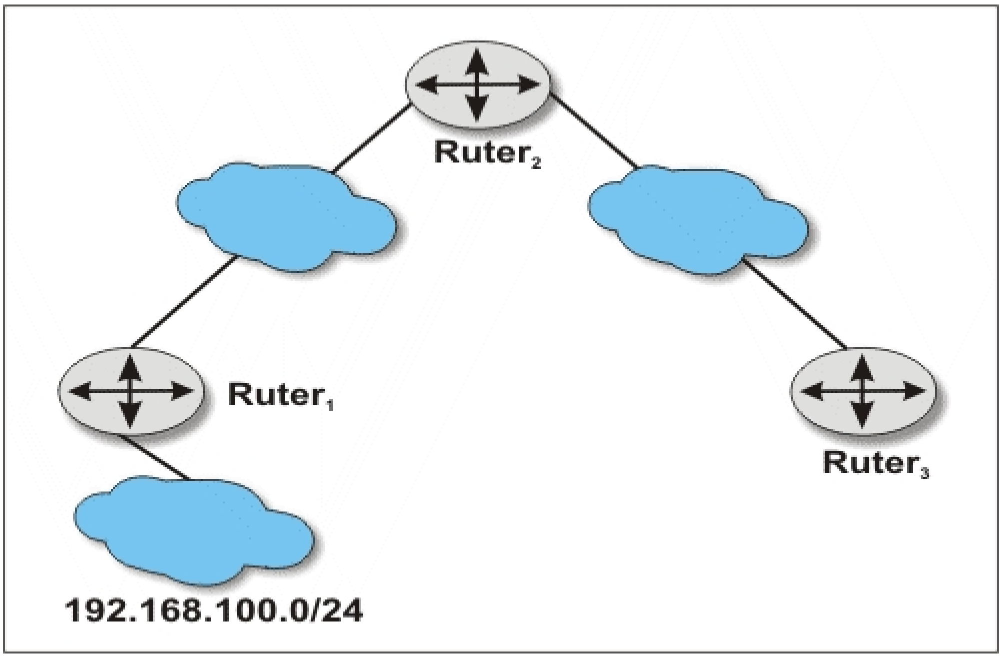
Protokół stanu łącza nie przekazuje informacji od ruterów o miejscach, które można za ich pośrednictwem osiągnąć. Zamiast tego przekazuje informację o topologii sieci. Informacja ta składa się z listy segmentów sieci lub łączy, do których dołączony jest dany ruter, oraz stanu tych łączy (czy funkcjonują, czy też nie). Informacje takie są następnie przepuszczane przez sieć. Przepuszczając te informacje coraz dalej w sieci każdy ruter jest w stanie zbudować sobie własny obraz sieci i bieżącego stanu wszystkich tworzących ją łączy. Ponieważ każdy ruter w sieci widzi te same informacje, wszystkie stworzone w opisany wyżej sposób obrazy sieci powinny być identyczne. Na podstawie takiego obrazu sieci każdy ruter wylicza najlepszą dla siebie trasę do poszczególnych miejsc w sieci i na tej podstawie tworzy tablicę rutowania. To, w jaki sposób ruter określa, która trasa jest najlepsza, zależy od algorytmu zastosowanego wdanym protokole. W najprostszych rozwiązaniach ruter może po prostu policzyć ścieżkę wykorzystując najmniejszą liczbę przeskoków. W bardziej złożonych protokołach informacje o stanie łącza mogą zawierać dodatkowe dane, które pomogą ruterowi określić najlepszą ścieżkę. Informacje takie mogą zawierać dane na temat pasma łącza, bieżącego obciążenia tego łącza, współczynników administracyjnych, a nawet ograniczenia przesyłania niektórych pakietów przez pewne łącza. Na przykład jakieś łącze w sieci może nie być wykorzystywane do przesyłania informacji tajnych.

Protokoły dystans-wektor oraz stanu łącza mają swoje dobre i złe strony. W poprawnie funkcjonującej i skonfigurowanej sieci każdy z tych protokołów poprawnie określi najlepszą trasę pomiędzy dwoma punktami.

Wady protokołów dystans-wektor.[2]

Ogólnie rzecz biorąc, protokoły typu dystans-wektor są łatwiejsze w konfigurowaniu niż protokoły stanu łącza. Łatwiej też zrozumieć ich działanie. W mniejszym stopniu obciążają one również procesor, co pozwala ruterowi zająć się innymi zadaniami, takimi jak przełączanie pakietów. Główne wady tych protokołów wynikają często z ich prostej budowy. Jedną z największych wad jest to, że rutery nie przekazują informacji o tym, skąd dowiedziały się o danej trasie, którą umieściły w komunikacie zawierającym uaktualnienia. Rozważmy np. prostą sieć zbudowaną z użyciem trzech ruterów, pokazaną na rysunku 1.5.1. $Ruter_1$ informuje $Ruter_2$ o sieci 192 .168 .100 .0/24. $Ruter_2$ będzie oczywiście informował $Router_2$ o tej sieci, ale taką samą informację może przekazać również do $Rutera_1$ $Router_2$ także może poinformować $Ruter_2$ o tym, że wie, jak dostać się do tej samej sieci, nawet jeśli trasa będzie prowadziła przez $Ruter_2$.

Rysunek 1.5.1: Prosta sieć złożona z trzech ruterów



Zwykle sytuacja taka nie jest problemem, ponieważ każdy ruter będzie porównywał miary tras, o jakich dowiaduje się z sieci, z miarami tras, które ma zapisane w tablicy rutowania, i na tej podstawie będzie wybierał najkorzystniejszą trasę. Co się jednak stanie, kiedy Ruter$_1$ straci połączenie z siecią 192.168.100.0/24 z powodu uszkodzenia sprzętu? Przestanie ona informować Ruter$_2$ o swoim istnieniu i w końcu zapis trasy do tej podsieci zostanie usunięty z tablicy rutowania tego rutera (trasa zostanie usunięta w wyniku upłynięcia określonego czasu lub na podstawie komunikatu przekazanego przez Ruter$_1$). Kiedy to nastąpi, Ruter$_2$ może usłyszeć od Routera$_2$ o istnieniu takiej sieci i doda tę „nową" podsieć do swojej tablicy rutowania, przekazując o tym informację Ruterowi$_1$. Oczywiście informacja wysłana zostanie również do Routera$_2$, który odkryje, że trasa prowadząca przez Ruter$_2$ jest gorsza od zapisanej poprzednio. Nie zważając na to, ruter uaktualni swoją tablicę rutowania i miarę, z którą będzie teraz rozgłaszał te informacje, wysyłając je do Rutera$_2$. Odebranie tego kolejnego uaktualnienia przez Ruter$_2$ spowoduje, że ogłosi on tę trasę (z trochę gorszą miarą) Ruterowi$_2$, który następnie zwróci informację do Rutera$_2$ z jeszcze większą miarą. W końcu rutery osiągną wartość miary, która jest zdefiniowana w danym protokole jako

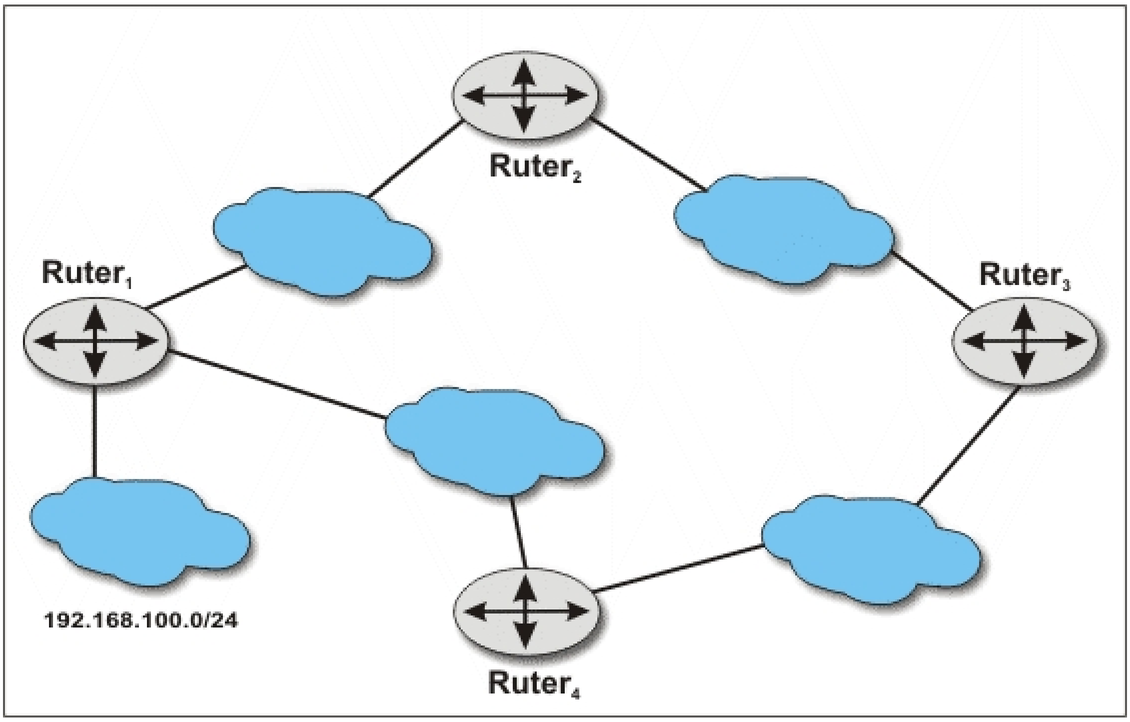
„nieskończoność”. Kiedy to się stanie, wszystkie rutery usuną tę trasę ze swoich tablic rutowania.

W zależności od tego, jak duża jest wartość „nieskończoności” określona w danym protokole, oraz od tego, jak często rutery wysyłają sobie uaktualnienia nawzajem, okres niestabilności i błędnego rutowania pakietów może trwać od kilku sekund do kilku minut. Niewątpliwie nie chcesz, aby tablice rutowania Twoich ruterów były niestabilne przez całe minuty za każdym razem, kiedy uszkodzeniu ulegnie jakaś część sieci! Większość protokołów typu dystans-wektor ma dodatkowe funkcje, które obsługują takie przypadki i zapobiegają przedłużaniu się czasu niestabilności. Pierwszą rzeczą, którą się zwykle dodaje, jest coś, co nazywane jest *domknięciem horyzontu.* W procedurze tej w momencie, kiedy ruter tworzy uaktualnienie dotyczące konkretnego interfejsu, pomija w nim wszystkie odniesienia do tras, których nauczył się od ruterów dostępnych przez ten interfejs.

W naszym przypadku oznacza to, że Ruter$_2$ poinformuje Router$_2$ o podsieci 192.168.100.0/24, której nauczył się z Rutera$_1$, ale pominie wszelkie odwołania do tej sieci, kiedy będzie wysyłał uaktualnienie do Rutera$_l$. Router$_2$ także powstrzyma się od informowania Rutera$_2$ o tej sieci, ponieważ to właśnie od tego rutera uzyskał o niej informacje. Niewielką modyfikacją metody „domkniętego horyzontu” jest metoda *„poison reverse„*. W metodzie tej zamiast pomijania informacji o sieciach, których ruter nauczył się z danego interfejsu, ruter dołącza te informacje do rozsyłanego uaktualnienia, ale dodaje do nich znacznik informujący, że taka sieć jest nieosiągalna. Taka informacja powoduje, że odbierający ją ruter posługujący się niewłaściwą trasą może ją usunąć ze swojej tablicy nitowania

Wynikiem działania w sieci opisanych wyżej metod jest fakt, że prosta niestabilność sieci opisana wcześniej nie może się w tej sieci zdarzyć. Niestety, ani jedna, ani druga metoda nie rozwiązują wszystkich problemów. Jeśli w sieci jest przejście łączące $Ruter_1$ z $Ruterem_2$, być może przez $Ruter_4$, jak pokazano na rysunku 1.5.2, możliwe jest wystąpienie pętli rutowania, nawet jeśli uruchomione są algorytmy opisanych wyżej metod. W takiej sieci $Ruter_l$ informuje $Ruter_2$ i $Ruter_4$ o swoim połączeniu z siecią 192 .168.100.0/24, podając w obu przypadkach prawdopodobnie taką samą wartość miary. Rutery te z kolei poinformują $Router_2$ o trasie prowadzącej do tej sieci, stosując prawdopodobnie tę samą miarę. $Router_2$ wybierze jedną z tych tras (prawdopodobnie tę, którą odbierze jako pierwszą) i umieści ją w swojej tablicy rutowania.

Rysunek 1.5.2: Standardowe metody nie zapobiegają występowaniu pętli rutowania w sieciach, w których połączenia tworzą pierścień



Załóżmy, że $Router_2$ wybierze trasę prowadzącą przez $Ruter_2$. Ponieważ działa metoda *„poison reverse„*, zgodnie z logiką działania tej metody wyśle on informację o tej trasie do $Rutera_2$ z miarą informującą o tym, że adres jest nieosiągalny. Ponieważ jednak zdecydował, że nie używa trasy prowadzącej przez $Ruter_4$, nie zastosuje wymienionej wyżej metody dla tego łącza, ale zamiast tego dołączy trasę do sieci

192.168.100.0/24 przez Ruter$_2$ do uaktualnienia wysyłanego do Rutera$_4$, który z kolei zignoruje to uaktualnienie i wybierze trasę prowadzącą przez Ruter$_1$.

Wszystko będzie działało dobrze do czasu, kiedy łącze pomiędzy Ruterem$_1$ i siecią 192.168.100.0/24 nie ulegnie uszkodzeniu. Wtedy Ruter$_1$ przestanie rozgłaszać tę trasę do Rutera$_2$ i Rutera$_4$. Rutery te z kolei przestaną rozgłaszać trasę do Routera$_2$, ale możliwe jest, że Ruter$_4$ usłyszy komunikat rozgłoszeniowy od Routera$_2$, zanim opisany wyżej proces dobiegnie końca. Ponieważ ruter ten nie ma informacji o trasie w swojej tablicy rutowania, umieści ją tam i poinformuje Ruter$_1$, że ma nową trasę. Następnie, zgodnie z działaniem algorytmu, Ruter$_1$ poinformuje o tej trasie Ruter$_2$, który prześle informację dalej, do Routera$_2$.

Pętla ta zostanie w końcu przerwana, kiedy każdy z ruterów, zwiększając miarę przy każdym przesyłaniu informacji o trasie w pętli, zwiększy ją do pewnej granicznej wartości dla danego protokołu, którą określaliśmy wcześniej jako wartość „nieskończoności”. Ile czasu zajmie ruterom tak zwane „odliczanie do nieskończoności”, zależy w dużym stopniu od tego, jak często wymieniają między sobą uaktualnienia, jaka jest wartość graniczna dla używanego w sieci protokołu i ile ruterów uczestniczy w tej pętli. Rozwiązaniem opisanego problemu jest wprowadzenie czasu blokowania. Kiedy ruter dowie się, że jakiś adres nie jest już dostępny dla ścieżki, której używał wcześniej, rozpoczyna odliczanie czasu, w trakcie którego ignoruje wszelkie inne informacje o lutowaniu dotyczące tego adresu. Czas ten wprowadzony jest po to, aby inne rutery mogły dowiedzieć się o wystąpieniu uszkodzenia, zanim ruter odliczający czas zacznie wykorzystywać ich trasy prowadzące do tego adresu docelowego. W naszym przypadku kiedy Ruter$_1$ stwierdza, że nie może dostać się do sieci 192.168.100.0/24, rozpoczyna odliczanie czasu blokowania tego zapisu w tablicy rutowania. W czasie odliczania ignoruje wszelkie uaktualnienia nadsyłane przez Router2. Jeśli czas blokowania jest wystarczająco długi, to zanim Ruter$_1$ zacznie znowu słuchać, Router2 stwierdzi, że jego trasa nie jest już popraw- na i nie będzie jej więcej rozgłaszał.

Wadą czasu blokowania jest to, że trudno jest określić, ile powinien on wynosić. Ile czasu zajmie rozpropagowanie informacji, że trasa nie jest już poprawna, do wszystkich ruterów, od których dany ruter otrzymuje uaktualnienia? Czasy te są szczególnie długie w przypadku protokołu takiego jak RIP. W swojej prostszej wersji RIP rozsyła uaktualnienia tablicy rutowania co 30 sekund. Ponieważ uaktualnienia te nie są potwierdzane przez odbiorców, możliwe, że niektóre z nich są gubione w sieci. Ponadto kiedy w uaktualnieniu znajduje się informacja o tym, jakie adresy są osiągalne, to nie zawsze wiadomo, które już osiągalne nie są. Nie ma więc żadnej wskazówki dla rutera, że powinien usunąć ze swojej tablicy rutowania trasę, która nie jest już dłużej poprawna.

Aby umożliwić wykrywanie zagubionych w sieci uaktualnień, RIP ustawia zegar dla każdej trasy, której się nauczył. Za każdym razem, kiedy RIP słyszy uaktualnienie dotyczące tej trasy, zegar jest zerowany. Jeśli ruter nie odbierze uaktualnienia w ciągu 180 sekund, usuwa trasę ze swojej tablicy rutowania i przestaje rozgłaszać ją swoim sąsiadom. W rezultacie jeśli jakieś uaktualnienie zostanie zagubione, rutery nie będą natychmiast usuwały tras ze swoich tablic rutowania. Prawdopodobnie trasy te znajdą się w kolejnym uaktualnieniu i ich zegary zostaną wyzerowane.

W praktyce procedura opisana wyżej oznacza, że rozgłoszenie zmiany w topologii sieci i zapisanie jej w tablicach rutowania wszystkich ruterów, które w tej sieci pracują, może zająć sporo czasu. Zastanów się jeszcze raz nad działaniem sieci z trzema ruterami, pokazanej na rysunku 1.5.1. Kiedy Ruter$_1$ stwierdzi, że utracił połączenie z siecią 192.168.100.0/24, to po prostu przestał rozgłaszać tę sieć w swoich uaktualnieniach wysyłanych do Rutera2. Mimo to przez kolejne 3 minuty od ostatniego komunikatu Ruter$_2$ nadal wierzył, że ma trasę prowadzącą do tej sieci i wysyłał informację o tym w uaktualnieniach kierowanych do Routera$_2$. Po trzech minutach Ruter$_2$ stwierdza, że Ruter$_1$ musiał utracić tę trasę i usuwa zapis trasy do sieci 192.168.100.0/24 ze swojej tablicy rutowania, informując o tym Router$_2$. Mimo to Ruter$_3$ nadal będzie wykorzystywał starą, nieaktualną już informację przez kolejne trzy minuty

Rozważmy teraz, co się będzie działo, jeśli taka procedura odliczania czasów na kolejnych ruterach wykonywana będzie w sieci składającej się z kilkunastu ruterów. Jeśli każdy z ruterów musi odczekać trzy minuty od czasu, kiedy najbliższy mu ruter przestał rozgłaszać daną trasę, to oczywiste staje się, że trasa może nie zniknąć całkowicie z sieci przez około 45 minut! Nierozsądne jest więc określanie tak długiego czasu blokowania rekordów w tablicy rutowania. Czas ten powinien stanowić niewielką część tych trzech minut. Aby redukować czas, kiedy w sieci występuje stan niespójności informacji o nitowaniu, protokół dystans-wektor umożliwia ruterom rozsyłanie informacji o *osiągalności negatywnej* dla tras, które zostały przez te rutery rozgłoszone, ale nie są już dłużej osiągalne. Informacje takie pozwalają ruterom na szybkie stwierdzenie faktu, że jakaś trasa nie jest dłużej dostępna. Dla protokołu RIP informacja o negatywnej osiągalności jest po prostu informacją o trasie z miarą ustawioną na wartość 16. Inne protokoły oznaczają taką informację we właściwy sobie sposób

Rozgłaszanie negatywne pomaga przyspieszyć przekazywanie informacji o uszkodzeniach tras, ale nie eliminuje opóźnień. Kiedy Ruter$_1$ odkryje, że jego połączenie z podsiecią 192.168.100.0/24 zostało przerwane (lub odtworzone), przekaże tę informację do Rutera$_2$ w kolejnym uaktualnieniu. W przypadku stosowania protokołu RIP jest to realizowane poprzez wysłanie uaktualnienia i może upłynąć do 30 sekund, zanim zostanie ono wygenerowane.

Ponadto jeśli $Ruter_2$ dostanie wiadomość od $Rutera_1$, to może również odczekać do 30 sekund, zanim powiadomi o zmianie $Router_2$, który z kolei odczeka do 30 sekund itd. Nawet jeśli informacja o zmianie stanu łącza przesłana zostanie przez sieć dość szybko, zwłaszcza w porównaniu z czasem, jaki jest potrzebny do wygaśnięcia zapisu w tablicy rutowania, to nadal może to zająć kilka minut, zanim wszystkie rutery w sieci dowiedzą się o tej zmianie i odpowiednio uaktualnią swoje tablice rutowania. Opóźnienie pomiędzy czasem wystąpienia zmiany stanu łącza w sieci a chwilą, kiedy wszystkie rutery w tej sieci dopasują swoje tablice rutowania, określane jest mianem *czasu zbieżności.* Długi czas zbieżności jest niewątpliwie problemem dla każdego protokołu rutowania

Aby zminimalizować czas konwergencji, protokół dystans-wektor może uruchomić wysyłanie uaktualnień *typu flash* lub *triggered.* Uaktualnienie *triggered* wysyłane jest za każdym razem, kiedy tablica nitowania danego rutera zmieni się w sposób, który może wpływać na rozsyłanie uaktualnień innych tras tego rutera. Jeśli każdy ruter używa uaktualnień tego typu i umieszcza w nich informacje o negatywnej osiągalności, to możliwe jest, że informacja o uszkodzeniu połączenia z $Rutera_l$ do sieci 192.168.100.0/24 zostanie przekazana do wszystkich ruterów pracujących w sieci w ciągu kilku sekund. Dzięki temu znacznie zmniejszy się czas zbieżności oraz czas, jaki ruter odczekuje przed usunięciem zapisu z tablicy rutowania.

Ten mechanizm nie jest prosty. Jeśli dodatkowe uaktualnienia nie będą dokładnie kontrolowane, to chwilowe uszkodzenie może powodować rozsyłanie w sieci tam i z powrotem różnych uaktualnień, co będzie zajmowało pasmo i moc obliczeniową procesorów w ruterach, które będą się zajmowały przetwarzaniem uaktualnień, a nie przełączaniem pakietów. Powszechnie stosowanym rozwiązaniem jest nieznaczne wydłużenie czasu odczekiwania przed usunięciem zapisu z tablicy rutowania oraz dodanie krótkiego czasu oczekiwania, który ustawiany jest po każdym uaktualnieniu typu *flash.* W czasie tego oczekiwania ruter nie przyjmuje żadnych innych uaktualnień, co pomaga złagodzić efekty faktycznych uszkodzeń

Kolejną dużą wadą protokołu typu dystans-wektor jest wada wynikająca z faktu, że nie jest to protokół zbyt skomplikowany. Ponieważ topologia sieci może ulec zmianie, w wyniku uszkodzenia łącza lub dodania albo usunięcia segmentu sieci, wszystkie dynamiczne protokoły rutowania muszą przekazywać do ruterów informacje o tych zmianach. W protokole dystans-wektor uaktualnienia wykonywane są zwykle po- przez okresowe rozsyłanie pakietów typu *broadcast (lub multicast*) poprzez niektóre lub wszystkie interfejsy rutera. Często uaktualnienia te zawierają pełną informację o nitowaniu, którą posiada ruter wysyłający to uaktualnienie. Okresowe uaktualnienia są przydatne, gdyż pozwalają ruterom pracującym w danym segmencie sieci informować się wzajemnie. Niestety, komunikaty te generują dodatkowy ruch w sieci nawet wtedy, kiedy sieć pracuje stabilnie (co, mamy nadzieję, stanowi większość czasu pracy sieci). Niektóre nowsze protokoły dystans-wektor, takie jak *Cisco EIGRP*, rozgłaszają tylko zmiany zachodzące w tablicach rutowania, ale protokół ten nadal jest rzadko stosowany.

Podczas gdy protokół dystans-wektor jest raczej nieskomplikowany oraz łatwy w obsłudze dla procesora rutera, prostota ta może prowadzić do nietypowych zachowań w wyniku uszkodzeń sieci i długich czasów zbieżności sieci. W sieci obsługiwanej przez ten protokół czas pomiędzy wystąpieniem uszkodzenia jednego z komponentów sieci a ustaleniem trasy obejściowej obsługiwanej przez poprawnie pracujące rutery może być dość długi. Działanie tego protokołu może również prowadzić do dużego wykorzystania pasma sieci i znacznego obciążenia procesora rutera nawet wtedy, gdy sieć pracuje stabilnie. Choć zmiany dokonywane w samym protokole mogą zmniejszyć te problemy, to po dodaniu dodatkowych funkcji rozgłaszania, obsługi czasów oczekiwania itd. protokół przestanie być zrozumiały i nieskomplikowany i znacznie trudniej będzie śledzić jego działanie.

Wady protokołów stanu łącza

Protokoły stanu łącza mają kilka ważnych zalet. Ponieważ obliczają trasy nitowania na podstawie znajomości topologii sieci, o której dowiadują się z uaktualnień informujących go o stanie łączy, nie mogą tworzyć pętli w wyniku częściowego uszkodzenia sieci, jak to zdarzało się w przypadku protokołów typu dystans-wektor. Ponieważ zmiany stanu łączy przekazywane są przez sieć natychmiast po ich wystąpieniu i docierają do wszystkich ruterów, które następnie uaktualniają swoje mapy topologii i tablice nitowania, to czas zbieżności sieci obsługiwanej przez taki protokół jest minimalny. Ostatnią zaletą, o której należy wspomnieć, jest fakt, że większość protokołów stanu łącza jest opracowana tak, by wysyłała uaktualnienia stanów łącza tylko wtedy, kiedy stan ten się zmieni, co sprawia, że protokoły tego typu oszczędzają pasmo i moc procesorów w czasie, kiedy sieć jest stabilna.

Choć protokoły stanu łącza zapobiegają powstawaniu pętli, skracają czasy zbieżności sieci i stopień wykorzystania zasobów sieci, mają też wady. Główną wadą jest ich złożoność. Złożoność jest głównym aspektem implementacji protokołu, ale często daje o sobie znać również podczas konfigurowania sprzętu. Tak naprawdę protokół OSPF, uważany za protokół wewnętrzny, jest znacznie bardziej skomplikowany od BGP, który stosowany jest jako protokół zewnętrzny. Na szczęście w typowej konfiguracji większość skomplikowanych funkcji ukryta jest przed użytkownikiem.

Dlaczego protokół stanu łącza jest tak złożony? Rozważmy jeszcze raz to, co mówiliśmy o sposobie, w jaki rutery określają swoje trasy. Zbierają one wszystkie uaktualnienia stanów łącza nadsyłane przez inne rutery i na ich podstawie budują mapę topologii sieci. Wykorzystując tę mapę rutery obliczają następnie najlepsze trasy do różnych miejsc w sieci. Pierwszym problemem jest generowanie mapy topologii. Choć człowiek może dość szybko narysować mapę połączeń sieci, bazując na informacjach o tym, co jest z czym połączone, to komputer musi mieć jakiś sposób zapisu tego ludzkiego rysunku w elektronicznej formie pozwalającej na dalsze przetwarzanie tych informacji. Standardowym sposobem zapisu tych informacji jest wykorzystanie jednego z wielu rodzajów grafów sieci.

Każdy rodzaj grafów ma pewien zestaw działań, które dobrze obsługuje, i zestaw funkcji, których nie obsługuje prawie wcale. Przeprowadzono wiele badań w celu opisania różnych typów grafów i funkcji, które te grafy obsługują. Bardzo często specyfikacja protokołu nie określa sposobu, w jaki ma być on implementowany. Możliwe, że w specyfikacji nie określa się nawet rodzajów danych, jakie będą konieczne do poprawnej pracy tego protokołu. Nawet jeśli rodzaje danych określone są w specyfikacji, to sposób w jaki dane te są reprezentowane (tzn. jaki rodzaj grafu zostanie użyty) pozostawia się temu, kto implementuje protokół. Zły wybór grafu może doprowadzić do trudno rozpoznawalnych uszkodzeń i błędów w kodzie oprogramowania rutera.

Drugą trudnością związaną z implementacją protokołu stanu łącza jest sposób liczenia najlepszej trasy do wszystkich miejsc w sieci. Choć istnieją algorytmy obliczające najlepszą ścieżkę za pomocą różnych typów grafów i miar, to nadal jest to kwestia odpowiedniej implementacji. Popełnione w procesie implementacji błędy dają ciekawe rezultaty w czasie działania produktu końcowego, jakim jest protokół rutowania w sieci.

Złożoność implementacji nie powinna być jednak przedmiotem zainteresowania administratora sieci, jeśli kod wynikowy, jaki otrzymał wraz z ruterami, działa poprawnie. Nawet jeśli kod jest poprawny, to skomplikowana implementacja wymaga zwykle większej mocy procesora i większej pamięci w ruterze. Na przykład wygenerowanie grafu topologii będzie zajmowało trochę czasu, a graf ten należy przecież jeszcze gdzieś zapisać. Musi on być przechowywany w dość bezpiecznym miejscu, ponieważ uaktualnienia stanu łączy zawierają tylko informacje o zmianach, jakie nastąpiły w topologii sieci. Dodatkowe wymagania dotyczące pamięci i mocy procesora sprawiają, że niektórzy administratorzy sieci trzymają się z dala od protokołów stanu łącza, ale nie jest to jedyny powód takiego postępowania. Ważniejszym powodem jest złożoność tych protokołów lub założenie, że są one skomplikowane i trudno je konfigurować.

Większość protokołów stanu łącza jest znacznie trudniejsza w konfiguracji niż protokoły typu dystans-wektor. Jeśli jednak interfejs konfiguracyjny jest dobrze zaimplementowany i jeśli zawiera zestaw właściwie określonych parametrów domyślnych, to możliwe jest skonfigurowanie protokołu stanu łącza przy niewiele większym nakładzie pracy niż dla protokołu dystans-wektor.

Zarówno protokół stanu łącza, jak protokół dystans-wektor będą działały poprawnie, jeśli rutowanie w stabilnej sieci będzie bezbłędne. Powinny one ponadto zmienić rutowanie na inne w sytuacji, kiedy w sieci wystąpi jakieś uszkodzenie.

---

[1] Matematyczna definicja wektora określa, że musi on mieć kierunek i długość. Niestety kiedy sieciowcy posługują się określeniem wektora w przypadku protokołów dystans-wektor, to myślą wyłącznie o jego kierunku.

[2] Scott M. Ballew *„Zarządzanie sieciami IP za pomocą ruterów CISCO”*, O’Reilly 1998r.

Sieci komputerowe

cztery punkty z pracy dyplomowej

Wprowadzenie

W ostatnich latach sieci komputerowe stały się niezbędnym narzędziem w przemyśle, bankowości, administracji, wojsku, nauce i innych działach gospodarki. Na rynku dostępne są różnorodne technologie sieciowe, których kierunki rozwoju określone są przez międzynarodowe organizacje standaryzacyjne i grupy robocze przy współudziale największych firm dostarczających sprzęt i oprogramowanie sieciowe.

Sieć komputerowa jest systemem komunikacyjnym służącym do przesyłania danych, łączącym, co najmniej dwa komputery i urządzenia peryferyjne.

Cel tworzenia sieci

Przyczyny zakładania sieci komputerowych i ich podstawowe cechy są następujące:

1. współużytkowanie programów i plików;
2. współużytkowanie innych zasobów: drukarek, ploterów, pamięci masowych, itd.
3. współużytkowanie baz danych;
4. ograniczenie wydatków na zakup stacji roboczych;
5. tworzenie grup roboczych – ludzie z różnych miejsc mogą uczestniczyć w tym samym projekcie;
6. poczta elektroniczna, szybkie i łatwe komunikowanie się;
7. oprogramowanie wspomagające pracę grup roboczych i obieg dokumentów;

rozwój organizacji – sieci mogą zmieniać strukturę organizacyjną firmy i sposób jej zarządzania;

Środowiska sieci

Środowisko sieci określone jest przez sieciowy system operacyjny oraz przez protokoły, zapewniające komunikację i usługi sieciowe. Istnieją 2 podstawowe typy sieciowych systemów operacyjnych:

1. ***każdy z każdym*** (*peer-to-peer*) – umożliwia użytkownikom udostępnienie zasobów swojego komputera oraz dostęp do zasobów innych komputerów. Wszystkie systemy w sieci mają taki sam status – żaden z nich nie jest podporządkowany innemu. Wszystkie stacje uczestniczące w sesji komunikacyjnej mają podobny stopień kontroli nad sesją, dysponują własną mocą przetwarzania i mogą kontrolować swoje działania. Rozwiązanie takie oferuje spore możliwości, nie jest jednak chętnie stosowane przez administratorów sieci ze względu na niewielkie możliwości zarządzania i niski poziom bezpieczeństwa. Występują tutaj problemy związane z lokalizacją danych, tworzeniem kopii zapasowych oraz z zapewnieniem odpowiedniej ochrony danych. Tworzenie sieci typu „każdy z każdym" umożliwiają m.in. systemy: IBM LAN Server, OS/2, LANtastic, Artisoft, MS Windows NT oraz MS Windows 95;
2. ***dedykowany serwer*** – jeden lub więcej komputerów spełnia rolę serwera i nie wykonuje innych zadań. Serwer spełnia takie zadania jak: przechowywanie i udostępnianie plików, zarządzanie współdzieleniem drukarek oraz funkcje związane z bezpieczeństwem danych;

Składniki sieci

Sieć komputerowa składa się zarówno ze sprzętu jak i z oprogramowania. Podstawowe składniki sieci to:

1. ***sieciowy system operacyjny***;
2. ***serwery*** – urządzenia lub oprogramowanie świadczące pewne usługi sieciowe, np.: serwer plików (przechowywanie i odzyskiwanie plików, włącznie z kontrolą praw dostępu i funkcjami związanymi z bezpieczeństwem), serwer poczty elektronicznej, serwer komunikacyjny (usługi połączeń z innymi systemami lub sieciami poprzez łącza sieci rozległej), serwer bazy danych, serwer archiwizujący, itd.
3. ***systemy klienta*** – węzły lub stacje robocze przyłączone do sieci przez karty sieciowe. System operacyjny klienta może zawierać oprogramowanie (powłoka – *requester*) skierowujące żądania sieciowe użytkowników lub aplikacji do serwerów;
4. ***karty sieciowe*** – adapter pozwalający na przyłączenie komputera do sieci. Stosowane są różne rodzaje kart w zależności od tego do pracy w jakiej sieci są przeznaczone;
5. ***system okablowania*** – medium transmisyjne łączące stacje robocze i serwery. W przypadku sieci bezprzewodowych może to być podczerwień lub kanały radiowe;
6. ***współdzielone zasoby i urządzenia peryferyjne – mogą to być drukarki, napędy dysków optycznych, plotery, itd.***

Translacja statyczna i dynamiczna

Kiedy mówimy o NAT to musimy wiedzieć, że translacje mogą być dokonywane statycznie (dokonywane ręcznie) lub dynamicznie. W pierwszym przypadku przydział adresu NAT-IP dla oryginalnego adresu IP jest jednoznaczny w drugim nie jest. W statycznym NAT pewien stały źródłowy adres IP jest zawsze translowany do tego samego adresu NAT-IP i żaden inny adres IP nie będzie translowany do tego samego adresu NAT-IP. Natomiast w przypadku translacji dynamicznej NAT, adres NAT-IP jest zależny od różnorodnych warunków działania i może być kompletnie inny dla każdej pojedynczej sesji.

Maski podsieci

Wszystkie adresy IP składają się z numeru sieci oraz numeru hosta w tej sieci. Jednakże granica pomiędzy numerem sieci a numerem hosta przebiega różnie w każdej z sieci. Aby można było w łatwy sposób określić gdzie ta granica leży, każdy z adresów ma dołączoną informacje w postaci maski podsieci. Jest to liczba podobnie jak adres IP 32 bitowa, gdzie wszystkie bity określające sieciową część adresu ustawione są na 1 a bity określające część adresu będącą numerem hosta ustawione są na 0.

Na przykład:

11111111 11111111 00000000 00000000

Oznacza to, że pierwsze 16 bitów adresu IP, z którym skojarzona jest ta maska reprezentuje adres sieci, natomiast pozostałe 16 bitów określa adres hosta w tej sieci.

Podobnie jak adres IP, maska sieciowa jest tradycyjnie reprezentowana przy użyciu zapisu kropkowo-dziesietnego lub szesnastkowego. A zatem maska może być zapisana jako:

255.255.254.0 lub jako 0xfffffe00.

W związku z tym, że obecnie maskę sieci zapisuje się jako nieprzerywalny ciąg bitów 1, możliwe jest posługiwanie się pojęciem maski 24 bitowej. Określenie to oznacza, że mamy do czynienia z maską gdzie wszystkie pierwsze 24 bity ustawione są na 1 a następne 8 na 0. Pozwala to adres 192.168.0.1 z maska 255.255.255.0 zapisać w postaci: 192.168.0.1/24 co nazywane jest zapisem w postaci adres/maska.

Podsieci i supersieci

W miarę jak protokół IP stawał się protokołem coraz częściej używanym przez administratorów sieci, zaczęto dochodzić do wniosku, że ustanowione na początku klasy sieci często nie odpowiadały na rzeczywiste potrzeby związane z zapotrzebowaniem na adresy IP. Często było tak, że marnowało się dużo adresów tylko dlatego, że każda z klas miała ściśle określoną rozpiętość adresową.

Np. Administrator potrzebując zaadresować 1200 hostów musi posłużyć się siecią z klasy B gdzie maksymalnie można zaadresować 65000 hostów, więc jest marnowana bardzo duża ilość adresów, tak szybko się wyczerpujących.

W związku z tym opracowano rozwiązanie podziału sieci na podsieci, gdzie po raz pierwszy tak naprawdę w pełni wykorzystano maski sieciowe.

Twórcy protokołu IP doszli do wniosku, że można wykorzystać bity w adresie IP opisujące numer hosta na podział sieci na mniejsze podsieci, tak, aby jak najefektywniej wykorzystać durze sieci z klasy A, B lub C.

Na przykład sieć z klasy A 10.0.0.0 jest opisana 8 pierwszymi bitami a następne 24 bity tworzą numer hosta. Można, więc podzielić tę sieć na mniejsze podsieci wykorzystując dodatkowe 8 bitów z części adresu opisującej numer hosta, które z adresu zostaną przypisane do adresu sieci. W ten sposób można stworzyć 256 podsieci, a w każdej z nich zaadresować 6500 hostów. Możliwe jest również wykorzystanie 16 bitów z numeru hosta dla określenia adresów podsieci, co zwiększa liczbę podsieci do 65000, a liczbę hostów w każdej z nich do 256.

Maska podsieci ma zawsze przynajmniej tyle bitów 1, ile jest ich w naturalnej masce dla danej klasy sieci. Oznacza to, że podsieć jest zawsze mniejsza od sieci, bez względu na to, z jakiej klasy ta sieć pochodzi. Kilka lat temu jak zaczęły się problemy z wyczerpywaniem się adresów IP zwrócono uwagę na fakt, że nie ma powodu, aby tak sztywno traktować maski sieciowe jak do tej pory. Dlaczego nie przydzielać sieci z maskami większymi niż naturalne dla klasy C i nie stworzyć bloków kilku sieci C traktowanych jako jedna sieć.

Takie rozwiązania są podstawą bezklasowego rutowania pomiędzy domenami (Classless Interdomain Routing – CIDR), które tworzy stosowaną obecnie w sieci architekturę bezklasową.