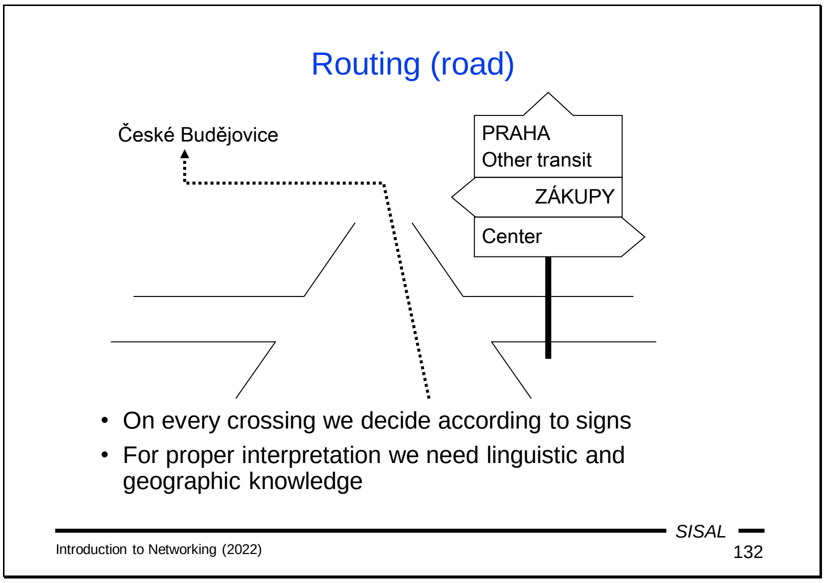
Slide 132



Routing (road)

České Budějovice

PRAHA
Other transit

ZÁKUPY

Center

- On every crossing we decide according to signs
- For proper interpretation we need linguistic and geographic knowledge

Introduction to Networking (2022)

SISAL

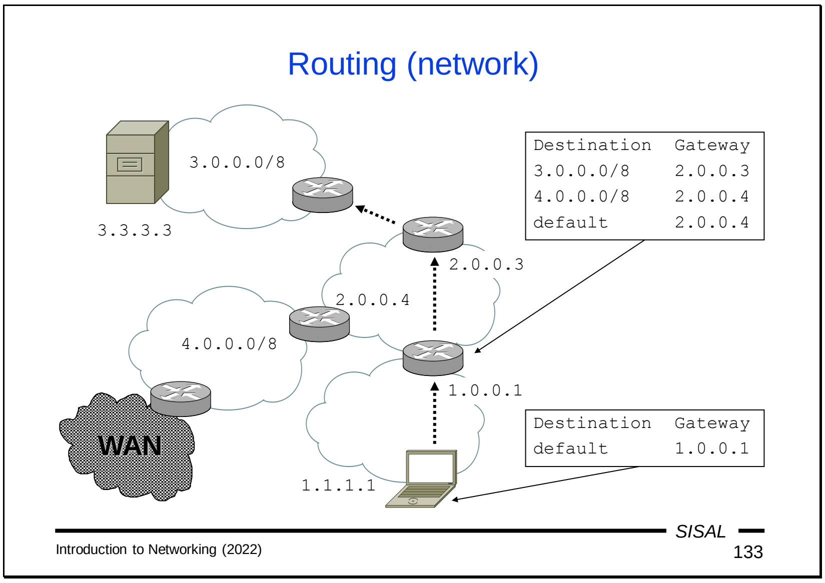
132

The issue of routing can be compared to a car driver’s decision at a crossroads. The driver has a specific destination in mind and uses direction signs to choose an exit from the intersection. The situation in the picture above schematically illustrates what a crossroads in Česká Lípa (a city in North Bohemia) might look like when arriving from the north.

Let's say that the driver's destination is České Budějovice (a city in South Bohemia) which isn't on the signs. The driver must therefore use geographical knowledge and conclude that if he is almost exactly north of Prague and his destination is almost exactly south of Prague, then it can be said that České Budějovice is "in the vicinity of Prague" and the road that leads toward Prague is a good choice for him. If he had not made this decision in Česká Lípa, but somewhere between Prague and České Budějovice, e.g. in Tábor, then choosing the road toward Prague would probably not have been correct. If the driver cannot choose based on the signs, he will have to use the direction "Other transit."

If the driver's destination is Zákupy, it is certainly true that Zákupy is somewhere around Prague, but since he has a far more accurate direction sign pointing toward Zákupy, he will follow this one and not the one towards Prague or for Other transit.

Slide 133



Routing (network)

Destination	Gateway
3.0.0.0/8	2.0.0.3
4.0.0.0/8	2.0.0.4
default	2.0.0.4

Destination	Gateway
default	1.0.0.1

Introduction to Networking (2022)

SISAL

133

Network software uses a similar process when it searches for the *next-hop* router on the path of a packet. It uses a *routing table* whose contents are controlled by the operating system. The records in this table are something like the direction signs at a crossroads. They also contain a “direction”, which in the computer world means a next-hop router (or *gateway*), and a *destination*. The advantage of a routing table is that each destination always contains a network address, including an address range specified by a network mask (*netmask*), so that here we are not dependent on any geographical knowledge and we know exactly which "surroundings" of the destination are relevant to us.

In the picture above we see a part of the network that our laptop with address 1.1.1.1 is connected to. We want to send a packet to address 3.3.3.3.

The typical configuration of an end station corresponds to the situation of a driver in a supermarket car park, where there's likely to be only one "Exit" sign. Similarly, on our laptop, there will be only one record in the routing table, a so-called *default record*, which will point to the router (1.0.0.1) through which our network is connected.

Once our packet arrives at the router 1.0.0.1, it will search for a next-hop in its routing table. The table will be a little more complicated here. We see three entries:

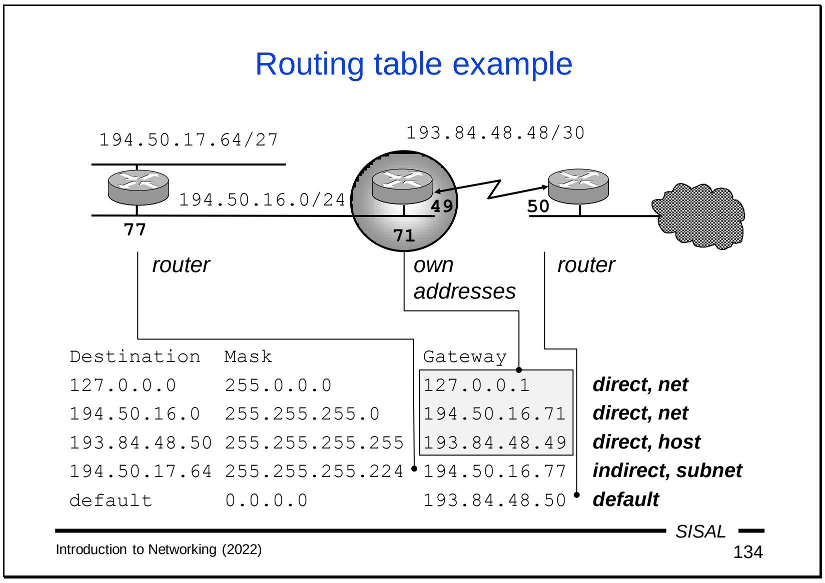
- The network with address 3.0.0.0 and /8 mask (i.e. a class A network) is reachable via router 2.0.0.3.
- The network 4.0.0.0 with mask /8 is reachable through router 2.0.0.4.
- To reach the other networks we can use the router 2.0.0.4 as well.

Which of these records fit our purpose? Just look at the mask for each record and compare the corresponding number of the first bits (the *prefix*) of the record address in the table to our destination address:

- The first record has a mask of 8 bits, so we compare the first byte of both addresses; both of them have value 3, so the record matches.
- For the second record, we also compare the first byte, but it does not match (4 is not equal to 3).
- The third record can be considered as having a mask of **0 bits**, so we compare 0 bits of both addresses and come back with a match.

We now have two suitable records and we have to choose from them. And this is the same situation as when a driver going to Zákupy chooses between the Zákupy and Prague signs – he chooses the direction that indicates a **smaller area** including his destination (an area around Prague that included Zákupy would have to be at least 200 km in diameter). A computer similarly chooses the record that has a smaller area i.e. a **wider mask**. So in our case, the correct option is the record 3.0.0.0/8 and not the default (which is actually 0.0.0.0/0).

Slide 134



Routing table example

Destination	Mask	Gateway	
127.0.0.0	255.0.0.0	127.0.0.1	*direct, net*
194.50.16.0	255.255.255.0	194.50.16.71	*direct, net*
193.84.48.50	255.255.255.255	193.84.48.49	*direct, host*
194.50.17.64	255.255.255.224	194.50.16.77	*indirect, subnet*
default	0.0.0.0	193.84.48.50	*default*

Introduction to Networking (2022)

SISAL

134

Now let's look at a routing table example in more detail. Let's imagine the situation from the picture above where we have a router connecting the LAN on the left via a point-point connection to the ISP router on the right.

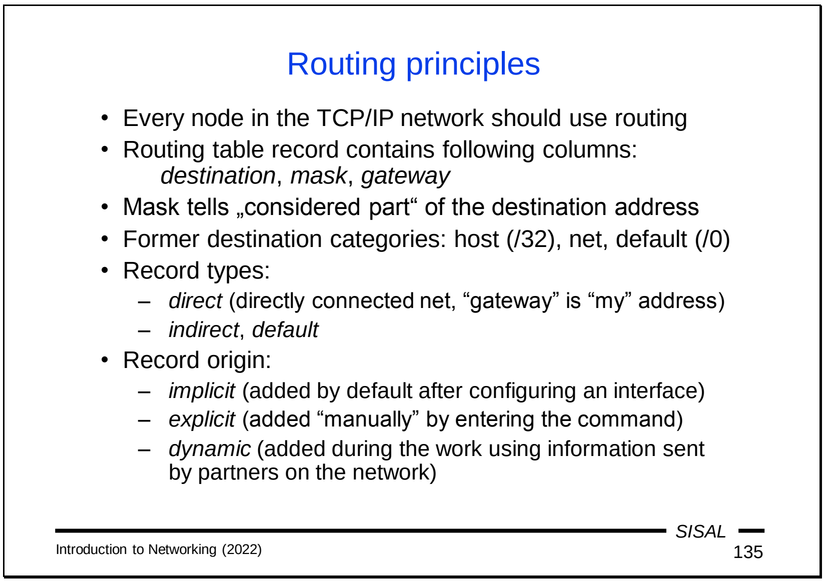
The routing table will contain three so-called **direct** records, or records that describe a directly connected network. These records are created automatically in the table when we configure the appropriate network interface.

- The first record describes a formal network interface with a loopback address. There is no next-hop router in the *gateway* column for direct records, but instead **our own address**, with which we are connected to the network (here 127.0.0.1). This makes good sense – when the software selects this record, it knows that it no longer has to search for a router, but can simply send the packet using a network interface with this particular address.
- The second entry describes the main network of our LAN with class C address 194.50.16.0. In the gateway column we have our own address 194.50.16.71 again.
- The third direct record is the record for the point-to-point network through which we are connected to the ISP. Here we can notice that the destination even has a /32 mask, i.e. one single machine, because at the end of the point-to-point connection there is only one machine, an ISP router with address 193.84.48.50.

Another entry is an **indirect** entry pointing to one of the subnets in our LAN, namely 194.50.17.64/27. An internal router in our network will be used as a gateway. We just need to be aware which of the addresses of this router will be in our table. The router, of course, has addresses on both networks 194.50.17.64/27 as well as 194.50.16.0/8. Of course, in our table, there must be an address from a network we know (ours), otherwise this record wouldn't be much help for us in routing.

The last record is a **default record** that directs all other traffic to the ISP router.

Slide 135



Routing principles

- Every node in the TCP/IP network should use routing
- Routing table record contains following columns:
  destination, *mask*, *gateway*
- Mask tells „considered part“ of the destination address
- Former destination categories: host (/32), net, default (/0)
- Record types:
  - *direct* (directly connected net, “gateway” is “my” address)
  - *indirect*, *default*
- Record origin:
  - *implicit* (added by default after configuring an interface)
  - *explicit* (added “manually” by entering the command)
  - *dynamic* (added during the work using information sent by partners on the network)

Introduction to Networking (2022) *SISAL* 135

Now let's review our findings on routing.

Any station operating on a TCP/IP network should be able to route. It compares a destination IP address in a packet with records in the routing table. These records contain three basic values:

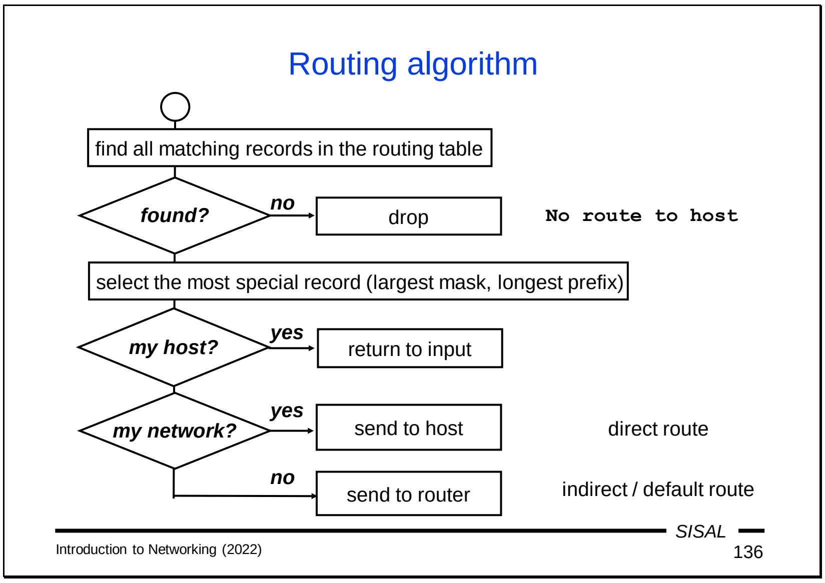
- the network address for which the record is valid (the *destination*),
- the range of this network (the *netmask*),
- a *gateway*, which is either
  - the next-hop router to which the packet needs to be forwarded if the record is indirect (leads to a foreign network),
  - the address of a custom network interface if the record is direct (leads to a directly connected network).

From the point of origin, we divide the records into these categories:

- implicit – a record is created automatically by configuring a network interface,
- explicit – a record is added to the table by a command that is either issued manually by the administrator or run by the operating system when the computer starts,
- dynamic – a record is created only during operation using information received from other nodes in the network.

Formerly, entries were categorized by their masks; an entry with a /32 mask was called a *host route*, and an entry with a narrower mask either a *net* or *subnet route*.

Slide 136



Routing algorithm

find all matching records in the routing table

found?

no

drop

`No route to host`

select the most special record (largest mask, longest prefix)

my host?

yes

return to input

my network?

yes

send to host

direct route

no

send to router

indirect / default route

SISAL

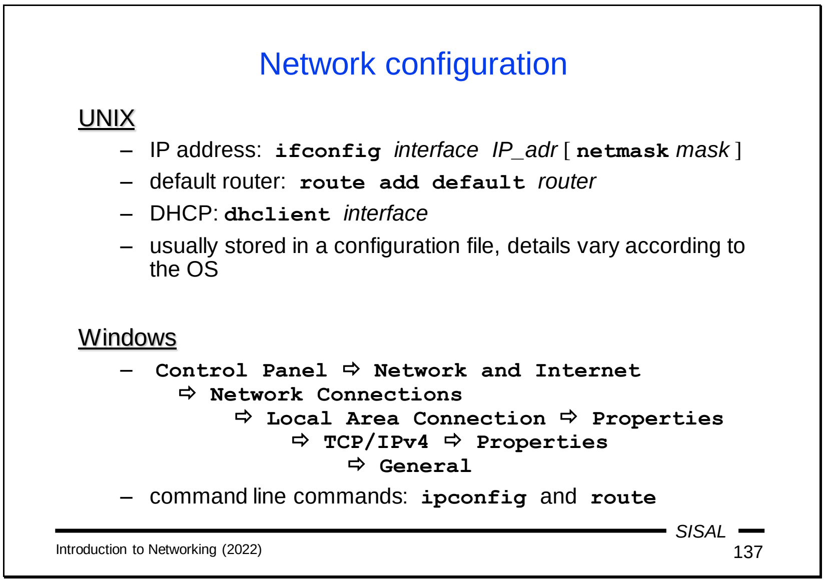
Introduction to Networking (2022)

136

We can describe the routing algorithm as follows:

- In the routing table, the algorithm searches for all records that match the packet’s destination.
- If no such record is found, the packet cannot be delivered. However, this can only be the case if the routing table does not contain a **default record** (that is, “Other transit” from the analogy with a crossroads). If the table contains one, at least this record will always match. If this situation occurs, the packet is **dropped**.
- The most specific record (the one with the widest mask) is chosen from the records found. The default record, if any, will therefore only be used if the table does not contain another record that matches the target.
- If the record references our own computer (loopback), the packet is placed on the input as if it had just arrived from the network.
- If the record is a direct one, the packet is directly passed to the link layer to be sent to the recipient.
- If the record is an indirect one, the packet is passed to the link layer with instructions to send it to the next-hop router.

Slide 137



Network configuration

UNIX

- IP address: **`ifconfig`** *interface IP_adr* [**`netmask`** *mask*]
- default router: **`route add default`** *router*
- DHCP: **`dhclient`** *interface*
- usually stored in a configuration file, details vary according to the OS

Windows

```
– Control Panel ⇨ Network and Internet
      ⇨ Network Connections
          ⇨ Local Area Connection ⇨ Properties
              ⇨ TCP/IPv4 ⇨ Properties
                  ⇨ General
```

- command line commands: **`ipconfig`** and **`route`**

SISAL

Introduction to Networking (2022) 137

Now we know the most important parameters to specify when configuring a TCP/IP network on a computer. So let's see how it's done.

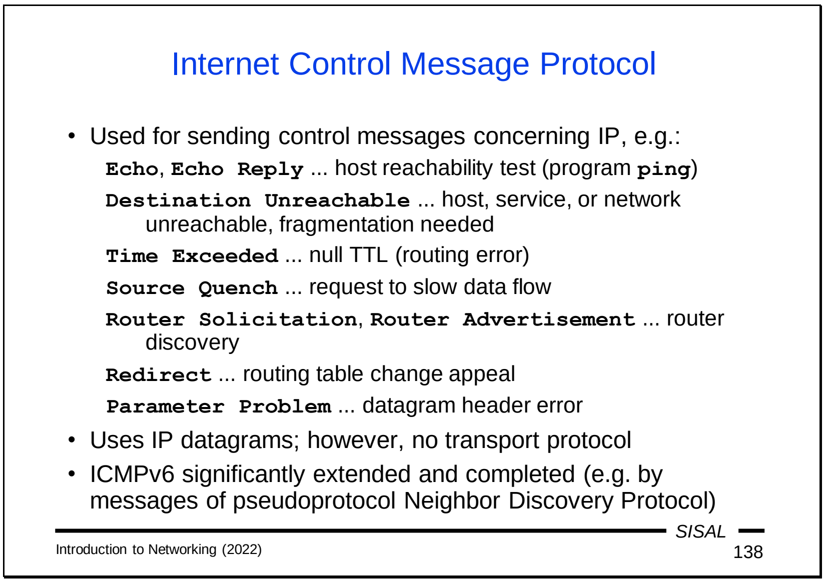
On UNIX systems, settings are controlled by the following commands:

- a network interface address is set with the **ifconfig** command,
- records in the routing table are changed by the **route** command,
- to perform the configuration via DHCP, you must run the **dhclient** command.

The commands are almost the same on all UNIX systems, but there are administrator tools for each of them that make configuration easier to manage, and these tools vary significantly from system to system.

On Microsoft Windows systems, network parameters are entered in a similar way to other settings, in older systems using the Control Panel, in newer ones using the Settings tool. However, there are also commands that can be entered from the command line (**ipconfig** and **route**).

Slide 138



Internet Control Message Protocol

- Used for sending control messages concerning IP, e.g.:
  - **`Echo`**, **`Echo Reply`** ... host reachability test (program **`ping`**)
  - **`Destination Unreachable`** ... host, service, or network unreachable, fragmentation needed
  - **`Time Exceeded`** ... null TTL (routing error)
  - **`Source Quench`** ... request to slow data flow
  - **`Router Solicitation`**, **`Router Advertisement`** ... router discovery
  - **`Redirect`** ... routing table change appeal
  - **`Parameter Problem`** ... datagram header error
- Uses IP datagrams; however, no transport protocol
- ICMPv6 significantly extended and completed (e.g. by messages of pseudoprotocol Neighbor Discovery Protocol)

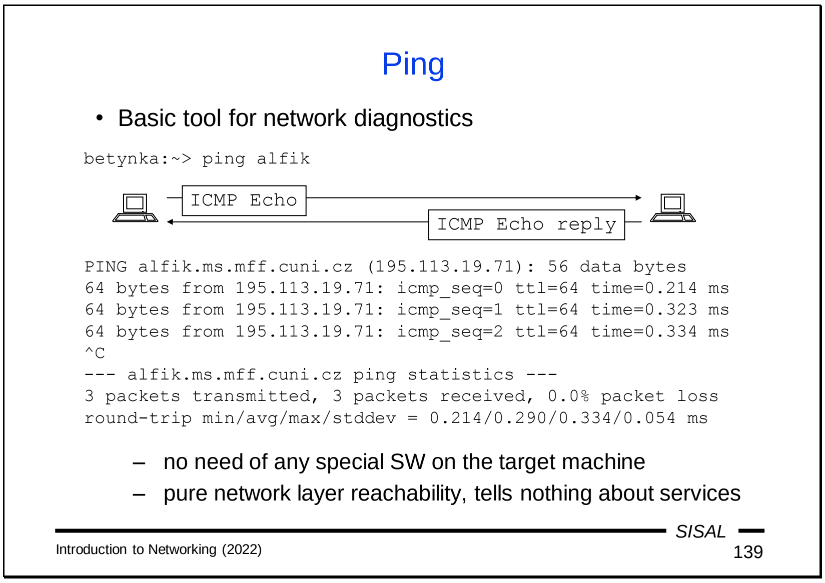
SISAL

Introduction to Networking (2022) 138

Before we continue, we need to learn one auxiliary protocol that IP networks use to send out information about different situations, which can then be used to better control the network. This is the Internet Control Message Protocol (ICMP). Its messages are sent as IP datagrams, so in the hierarchy, it can be placed above the third layer of the OSI model, but it cannot be placed at the fourth layer as a transport protocol. IP version 6 has significantly expanded the functionality of ICMP, e.g. it distributes Neighbor Discovery Protocol (NDP) messages that a station can use for detecting the status of the surrounding network.

Further on, we will talk about the most important types of messages, so I will only mention the **Destination Unreachable** message at this point. With this message, a station or router announces that it has no way to deliver a received packet and that it discards it. In the case of a router, this is usually due to a failure to find a matching record in the routing table. In the case of a station, this message most often means that there was an attempt to deliver data to a UDP service that is not available (in the case of TCP, such an attempt is resolved using a response with the RST flag at the transport protocol level, but there is no such option in UDP, so this is handled by ICMP).

Slide 139



Ping

- Basic tool for network diagnostics

```
betynka:~> ping alfik
```

ICMP Echo

ICMP Echo reply

```
PING alfik.ms.mff.cuni.cz (195.113.19.71): 56 data bytes
64 bytes from 195.113.19.71: icmp_seq=0 ttl=64 time=0.214 ms
64 bytes from 195.113.19.71: icmp_seq=1 ttl=64 time=0.323 ms
64 bytes from 195.113.19.71: icmp_seq=2 ttl=64 time=0.334 ms
^C
--- alfik.ms.mff.cuni.cz ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.214/0.290/0.334/0.054 ms
```

– no need of any special SW on the target machine
– pure network layer reachability, tells nothing about services

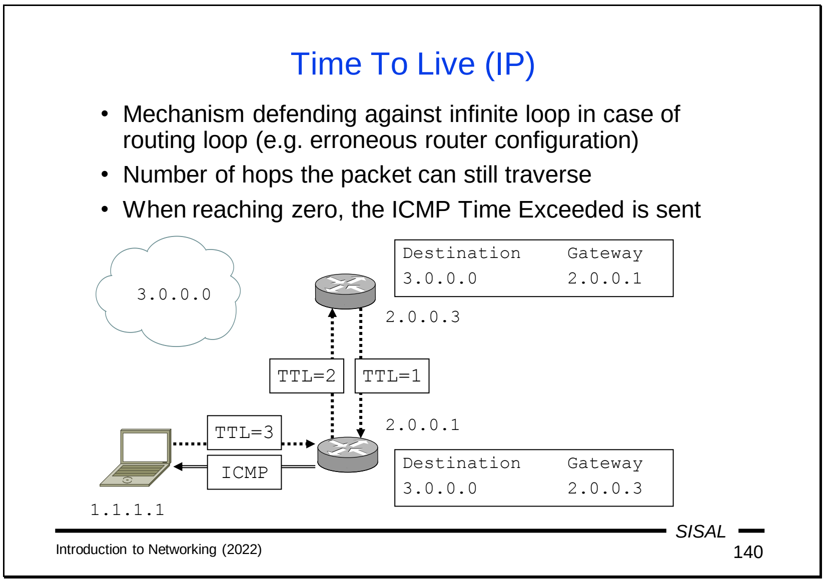
Introduction to Networking (2022) SISAL 139

The **ICMP Echo** and **ICMP Echo reply** messages are used by the **ping** program, an essential tool for network diagnostics. Using it, we can test the accessibility of a remote network node.

When we run the ping program, it will begin to periodically broadcast ICMP Echo messages to a specified destination machine. When each of these messages reaches the target, its network software will respond with an ICMP Echo Reply. When the answer comes back, ping writes a line with information about how long it took to arrive. The program broadcasts messages with a period of 1 second, until we interrupt it or it has sent a given number of messages. It then prints statistics, i.e. the number and percentage of responses received and the minimum, maximum, average and standard deviation of a value called the *round-trip time* (bidirectional delay).

The important fact is that no special server has to be running on the target machine, ICMP handling is the responsibility of the network software itself. By using ping, however, we will only verify that our packets can travel to and from the target, not that the target will be willing to provide any application services. In reality, the opposite may even be the case: we may be able to make an HTTP connection with a computer even though a ping attempt failed – either the target machine may have ICMP Echo responses disabled in its configuration, or ICMP Echo/Echo reply packets may be withdrawn by some router along the way.

Slide 140



Time To Live (IP)

- Mechanism defending against infinite loop in case of routing loop (e.g. erroneous router configuration)
- Number of hops the packet can still traverse
- When reaching zero, the ICMP Time Exceeded is sent

Destination	Gateway
3.0.0.0	2.0.0.1

Destination	Gateway
3.0.0.0	2.0.0.3

Introduction to Networking (2022)

SISAL

140

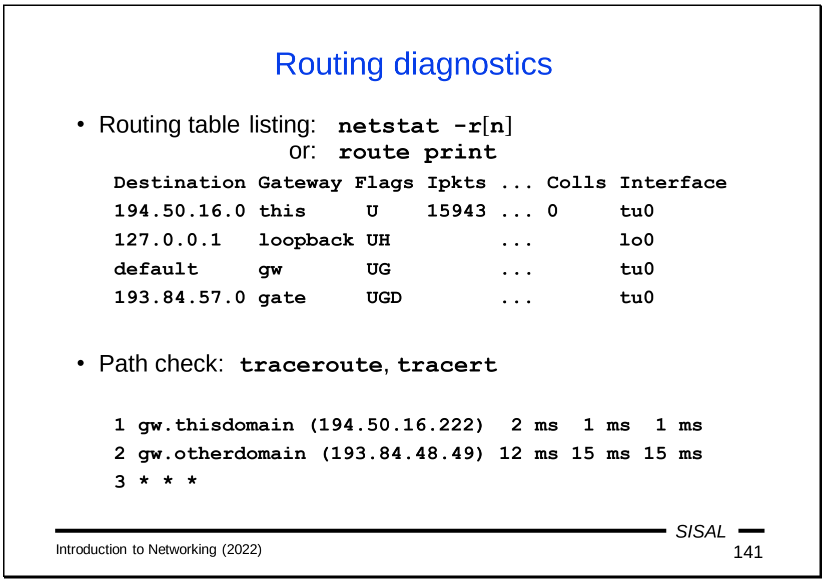
Another important ICMP message is **Time Exceeded**. It is related to the Time-to-live (**TTL**) field in the IP header, and together they are used to prevent packets from running in an infinite loop due to an error in the routing tables.

The situation in the image above resulted in a routing loop between routers 2.0.0.1 and 2.0.0.3 on the way to network 3.0.0.0. Both routers have a record for this network in their routing tables, pointing to each other. This can occur either through an administrator mistake or a software bug. The TTL will prevent this loop from having fatal consequences. It expresses the **number of routers** allowed to forward the packet (the number of "hops").

Imagine that our laptop is about to send a packet to address 3.3.3.3 and that its software sets the packet's TTL to 3 (this setting can be changed by a special function call; the usual default is 64, today). The packet arrives at router 2.0.0.1, which finds that the packet should be forwarded to router 2.0.0.3. And so it decrements the TTL to 2, and since this value is not zero, it actually forwards the packet. A similar procedure will take place on router 2.0.0.3 and the packet will return to router 2.0.0.1 with TTL 1. But now decrementing results in a zero value, so the packet will not be forwarded again; instead, the router **drops** the packet, and sends an ICMP Time Exceeded message to the original sender.

Note 1: Note that in this context, the terms "Time Exceeded" and "Time-to-live" are really **not about time**, but the number of hops! The name is historical. On the other hand, in DNS records, a TTL value really means a time in seconds.

Note 2: **Every** router must lower the TTL value and thereby **modify** the content of the IP header, forcing the router to recalculate the **checksum**. In IPv6, the header checksum was therefore dropped.



Routing diagnostics

- Routing table listing: **netstat -r**[**n**]
  or: **route print**

```
Destination Gateway Flags Ipkts ... Colls Interface
194.50.16.0 this     U    15943 ... 0     tu0
127.0.0.1   loopback UH         ...       lo0
default     gw       UG         ...       tu0
193.84.57.0 gate     UGD        ...       tu0
```

- Path check: **traceroute**, **tracert**

```
1 gw.thisdomain (194.50.16.222)  2 ms  1 ms  1 ms
2 gw.otherdomain (193.84.48.49) 12 ms 15 ms 15 ms
3 * * *
```

SISAL

Introduction to Networking (2022)

141

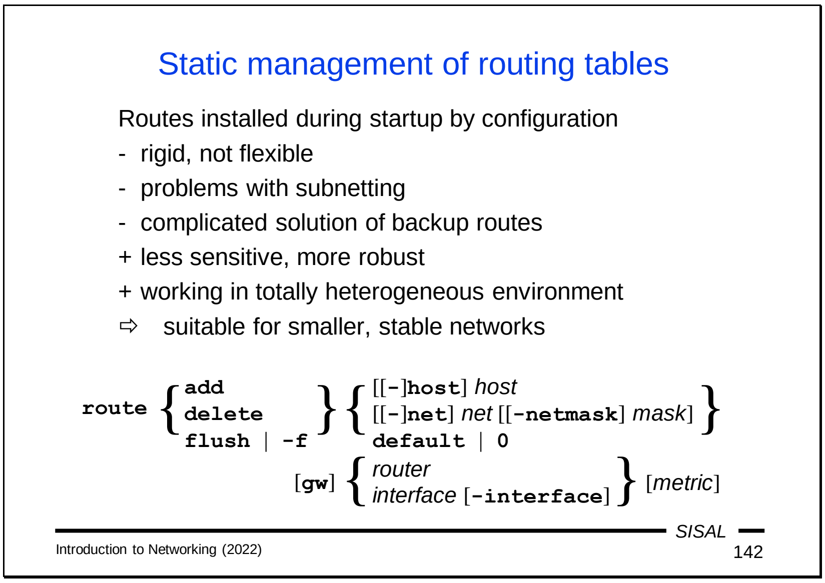
If we need to diagnose routing problems, we usually start by listing a routing table. This can be triggered by a **netstat –r** or **route** print command. In addition to the columns we've already seen on previous slides, the listing above also shows flags (H for *host route*, G for *gateway*, i.e. indirect record, D for *dynamic* record), the name of the network interface, and various statistics (which give the netstat command its name).

Note: The **-n** option prevents the netstat program from trying to translate IP addresses into names. There are two reasons for this: first, we tend to be more interested in numerical values, and second, with this option we do not run the risk that a broken network will fail to translate an address and we will learn nothing.

The next diagnostic step is likely to be **ping**, but it usually doesn't help. In fact, if an answer is not returned, there might be various reasons for this and we may not be able to distinguish them. First, our packets may not have reached their destination. Second, they may have arrived, but the target may not be functional. But it is also possible that the packets were lost on the way back. The routing tables work **in one direction**! It is again similar to our car example – the fact that we are able to travel from Česká Lípa to České Budějovice by following direction signs doesn't mean anything about whether we'll find the way back.

A better diagnostic tool is the **traceroute** command, which exploits the properties of the TTL field. The program first sends a packet with TTL 1 to the destination. As a result, if the target lies behind at least one router, the packet won't reach the destination because the router will refuse to forward it and will send back an ICMP Time Exceeded message. The traceroute program will capture it and display the address (and possibly the name) of the router that sent it, including the round-trip time. The program performs this attempt three times and then starts sending exactly the same packets, but with the TTL increased to 2. These packets now pass through the first router, but are stopped by the following hop. In this way, the program will gradually disclose the structure of the path to the destination. The process will either end in success, or responses will stop coming back at a certain TTL value. The last router that responded to us will then be the starting point of our investigation – either the error is directly at it, or at the next router, or at the previous one (in case the last responding router should never have received our packets).

Slide 142



Static management of routing tables

Routes installed during startup by configuration

- rigid, not flexible
- problems with subnetting
- complicated solution of backup routes

+ less sensitive, more robust

+ working in totally heterogeneous environment

⇨ suitable for smaller, stable networks

```
route { add          } { [[-]host] host                  }
      { delete       } { [[-]net] net [[-netmask] mask]  }
      { flush | -f   } { default | 0                     }
                  [gw] { router                  } [metric]
                       { interface [-interface]  }
```

Introduction to Networking (2022)

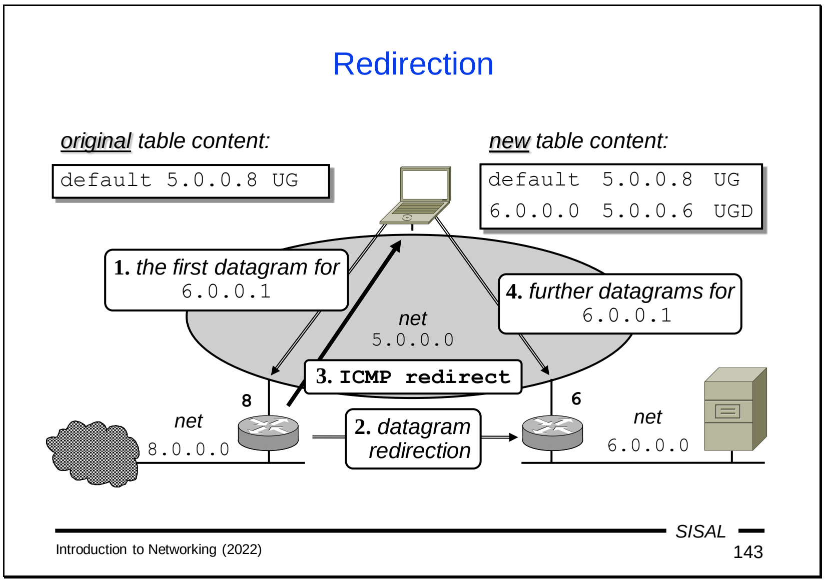
SISAL

142

Now we know how a computer uses a routing table, but we don't yet know how the content of the table is managed.

The simplest approach is the **static method** of table management, in which a computer has data for all the needed records stored somewhere, and adds them to the table one by one after it boots. The advantage of this method is its stability – if the network does not change often and is not too complex, this method guarantees proper functioning in all circumstances and with different types of nodes. This method is used, by the way, when we plug our computer into a network and it gets the address of the **default router** of the local network through DHCP. That's all it needs for routing. But if the network is too large or changes dynamically, this approach is not appropriate.

Slide 143



Redirection

original table content:

```
default 5.0.0.8 UG
```

new table content:

```
default  5.0.0.8  UG
6.0.0.0  5.0.0.6  UGD
```

1. *the first datagram for* 6.0.0.1
net 5.0.0.0
3. ICMP redirect
4. *further datagrams for* 6.0.0.1
8
6
net 8.0.0.0
2. *datagram redirection*
net 6.0.0.0

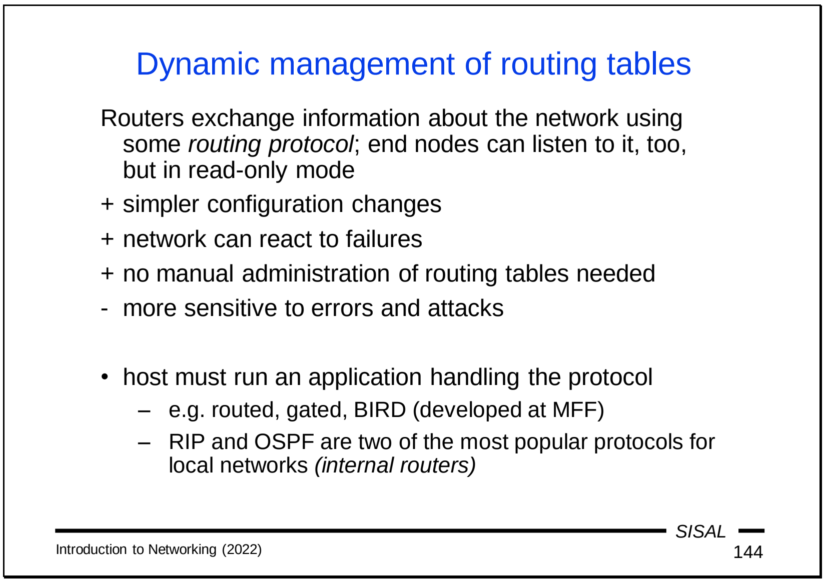
One way to cover even a more complex network, with static routing tables, is to use another type of ICMP message, namely **ICMP Redirect**.

In the LAN in the image above we see network 5.0.0.0 with the default router 5.0.0.8. This default route is distributed to every station, so we can see it in the routing table of our laptop. But there's another network 6.0.0.0 plugged into the LAN that our laptop doesn't know about yet. What happens if we try to send a packet to this network?

- Following its routing table, our laptop sends the packet to router 5.0.0.8.
- The router accepts the packet and is going to forward it. But it finds that it has to send the packet **back to the same** network it came from, only to another router 5.0.0.6. This indicates an error on the sender's side, as it could have sent the packet to router 5.0.0.6 directly. So router 5.0.0.8 redirects the packet to the correct router, but...
- …it also sends the original sender an ICMP Redirect message telling it to add a new record for network 6.0.0.0 to its routing table.
- If our laptop makes the change, subsequent packets for this network will now be sent to the correct router. In the routing table, we can then see a new record for this network, with a D flag (for a dynamically added record).

While ICMP Redirect solves other similar situations, it may also cause problems. If someone (intentionally or by software error) disseminates defective ICMP Redirect packets on the local network, they may cause the network to malfunction. It is therefore usually better to distribute the full network structure to clients and ban the reception of ICMP Redirect.

Slide 144



Dynamic management of routing tables

Routers exchange information about the network using some *routing protocol*; end nodes can listen to it, too, but in read-only mode

+ simpler configuration changes
+ network can react to failures
+ no manual administration of routing tables needed
- more sensitive to errors and attacks

- host must run an application handling the protocol
  - e.g. routed, gated, BIRD (developed at MFF)
  - RIP and OSPF are two of the most popular protocols for local networks *(internal routers)*

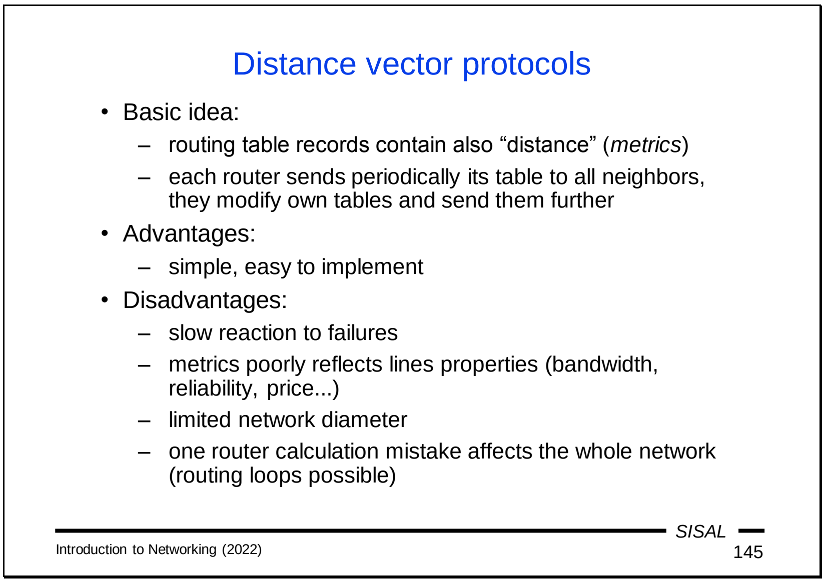
The solution for more complex networks is a **dynamic** method to manage tables. This is based on the information exchanged between neighboring routers used to adjust their routing tables. The routers communicate with each other using one of the **routing protocols**. Ordinary stations can also follow this information, but only in passive mode.

A network so controlled can adapt itself to current conditions, and configuration changes can be made centrally in one place. A small cost of this approach is a certain load on the network from routing protocol messages and a greater vulnerability of the network to errors caused by software attacks or errors.

If a node (router or station) wants to be involved in routing protocol communication, special software must be running on it. One highly successful routing program, **BIRD**, which is now used on a number of key routers in the world, was originally developed as a **student project at our faculty**.

Routing protocols for local networks are divided into distance-vector protocols (e.g., RIP) and link-state protocols (e.g., OSPF).

Slide 145



Distance vector protocols

- Basic idea:
  - routing table records contain also “distance” (*metrics*)
  - each router sends periodically its table to all neighbors, they modify own tables and send them further
- Advantages:
  - simple, easy to implement
- Disadvantages:
  - slow reaction to failures
  - metrics poorly reflects lines properties (bandwidth, reliability, price...)
  - limited network diameter
  - one router calculation mistake affects the whole network (routing loops possible)

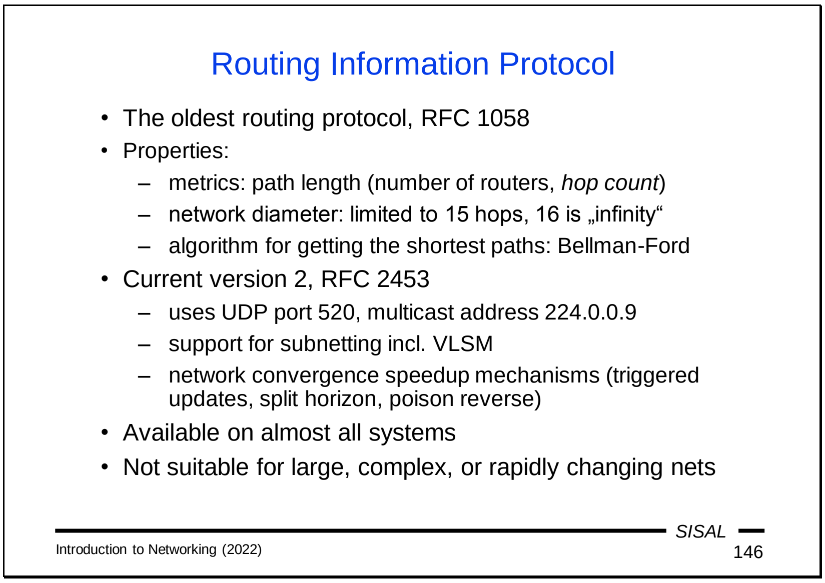
The basic idea of **distance-vector** protocols is something we can easily imagine in our crossroads example again. Direction signs also contain distances in kilometers, so if we regularly send photos of our crossroad signs to neighboring crossroads, administrators there can easily determine whether there is a path to a given destination via our crossroads, better than the one they have on their signs now. If so, they simply turn the sign in the right direction and write a new distance on it.

And it works similarly with routers – they periodically send their routing tables to their neighbors, who check to see if they can change any of the records in their routing tables.

The advantage of these protocols is relative simplicity and easy implementation.

However, there are some drawbacks such as a slow response to errors, an insufficiently fine evaluation of individual paths (we know only the number of kilometers to a given target, but not the quality of the path) and limited network range. But the absolutely fundamental problem is that if a router makes a **miscalculation**, it will **spread** it and may render the entire network inoperable.

Slide 146



Routing Information Protocol

- The oldest routing protocol, RFC 1058
- Properties:
  - metrics: path length (number of routers, *hop count*)
  - network diameter: limited to 15 hops, 16 is „infinity“
  - algorithm for getting the shortest paths: Bellman-Ford
- Current version 2, RFC 2453
  - uses UDP port 520, multicast address 224.0.0.9
  - support for subnetting incl. VLSM
  - network convergence speedup mechanisms (triggered updates, split horizon, poison reverse)
- Available on almost all systems
- Not suitable for large, complex, or rapidly changing nets

SISAL

Introduction to Networking (2022) 146

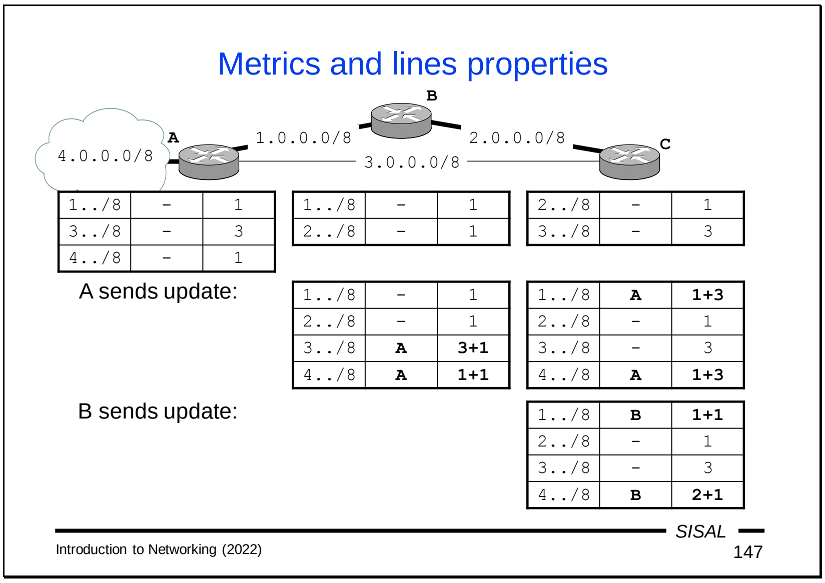
The best-known distance-vector protocol is RIP (Routing Information Protocol). It belongs among the oldest protocols, which makes its implementation very widespread, which is its main advantage.

At the same time, the era of its creation affected some of the decisions in the proposal. The role of **metrics** ("distance") is played here by the number of routers (hops) on the way to the destination. However, that number is limited to only 15 in the proposal, with unpleasant consequences that we will discuss later. These consequences are partially eliminated by the extensions added in version 2. Despite this, RIP is not suitable for large networks or networks, the structure of which often changes.

In every article about RIP, you'll find a note that it uses the **Bellman-Ford** algorithm to calculate the shortest paths, but there's almost no explanation for why that's the case and why Dijkstra's algorithm which is generally faster, isn't used instead. The reason is that by its very nature the Bellman-Ford algorithm actually "replicates" the incremental correction of calculated distances based on new information from neighboring nodes. Simply said, the arrival of a new message from our neighbor's routing table will trigger the calculation of a **single** step of the Bellman-Ford algorithm, whereas with Dijkstra’s algorithm we would have to repeat the **whole** calculation.

Interestingly, version 1 of RIP did not support variable length masks (VLSM) networks because no mask was included in the packet format. Version 2 now includes a mask in the format, despite the fact that the packets have not lengthened in any way and are backward compatible – space that was unused in the version 1 format was simply used for it.

Slide 147



Metrics and lines properties

1../8	-	1
3../8	-	3
4../8	-	1

1../8	-	1
2../8	-	1

2../8	-	1
3../8	-	3

A sends update:

1../8	-	1
2../8	-	1
3../8	**A**	**3+1**
4../8	**A**	**1+1**

1../8	**A**	**1+3**
2../8	-	1
3../8	-	3
4../8	**A**	**1+3**

B sends update:

1../8	**B**	**1+1**
2../8	-	1
3../8	-	3
4../8	**B**	**2+1**

Introduction to Networking (2022)

SISAL

147

The number of hops along a path, unfortunately, does not reflect the properties of individual segments well. It is again similar to the situation on a road – if we can reach a city by travelling 80 km by road or 100 km by highway, the longer journey will be quicker, but if the driver only decides based on the number of kilometers, he will take the slower route. In the picture above, routers A and C are linked by a direct line (the number of hops between them is therefore 0) which is "slow“, and also by a "fast" line which runs through router B and therefore has a metric with one hop more. RIP will therefore prefer the slower line with a metric of 0. This can be prevented by artificially penalizing the slow line with higher metrics than the number of hops would normally indicate. In our example, we have "added two routers" to the line.

Now let's see how the network situation will change as the routers exchange information (send updates of their tables).

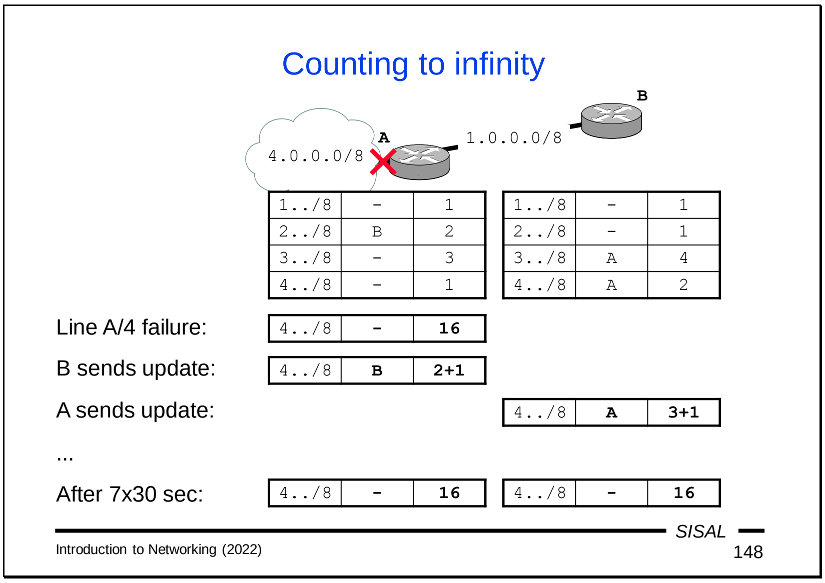
- In the beginning, all routers have initialized their tables with values for directly connected networks. These will all have a metric of 1, except for the line between A and C, which will have its metric artificially raised to a value of 3.
- Let's say that the first update is sent by router A. The table arrives at both routers B and C, which add records for paths to the new networks included in the RIP messages. When calculating a metric for the new networks, they take the metric in the message and add a metric to the network from which the packet arrived. Thus, router B will add 1 to all the values, so it will add network 3.0.0.0 with a metric of 3+1 and network 4.0.0.0 with a metric of 1+1 to its table. Router C has its metric to the network 3.0.0.0 (from which the message arrived) set to 3 and will therefore add both new records 1.0.0.0 and 4.0.0.0 with metrics of 1+3. The directly connected networks of both routers have a metric better than the one that was received, so those won't change.

- Next, router B sends its update. Router C will receive its packet over a network that has a metric of 1, so it will add one to the values in the message. Network 1.0.0.0 had a reported metric of 1, adding one gives us a value of 2, which is better than the current value of 4. So router C modifies the record in the table, changing the direction to router B and metric to 2. Check of network 4.0.0.0 has a similar result: the new metric 2+1 is again better than the existing 4, and the record is corrected. Directly connected networks remain unchanged. This form of the table on router C is now final, unless there is some change in the network.
- The process concludes with an update sent by router C, which allows the remaining routers to finish changes in their tables.

Since routers send initial messages with the contents of their tables right after they boot, this network converges very quickly.

We have solved the problem of varying line quality elegantly here, but only because this network is very small and the lines have only two different "speeds" (bandwidths). Once we have a more complex network, we will very quickly hit the ceiling of the maximum metric of 15. The logical solution would be to fundamentally raise that value.

Slide 148



Counting to infinity

1../8	-	1	1../8	-	1
2../8	B	2	2../8	-	1
3../8	-	3	3../8	A	4
4../8	-	1	4../8	A	2

Line A/4 failure:	4../8	**-**	**16**			
B sends update:	4../8	**B**	**2+1**			
A sends update:				4../8	**A**	**3+1**
...						
After 7x30 sec:	4../8	**-**	**16**	4../8	**-**	**16**

Introduction to Networking (2022) *SISAL* 148

Why is RIP's "infinity" so small? The reason is to try to minimize the impact of a problem called **counting to infinity**.

Let's imagine that there's a connection failure to network 4.0.0.0 on router A. The router responds by setting this network's metric to infinity (16). But before its period elapses to send out its routing table update to distribute information about the unavailability of the network 4.0.0.0, it receives an update packet from router B, which still has the network with metric 2 in its tables. Router A makes a calculation and finds that the "new" route to network 4.0.0.0 via router B has a metric of 2+1 < 16, and so “corrects” it in its table. This will remove information about the unavailability of the network.

Once the period for router A's update elapses and it sends its table to everyone, router B will be informed that network 4.0.0.0 is now available via router A, but with a metric of 3. Since the existing entry on router B also originates from router A, router B must respect the change in the metric on router A, and correct the network 4.0.0.0 record to the new metric of 3+1.

With subsequent exchanges of update packets, the metric on both routers gradually increases until it finally returns to the correct "infinity" value. Due to the length of the period (30 sec), for over 3 minutes the network 4.0.0.0 appears to be alive, although it is not really available.

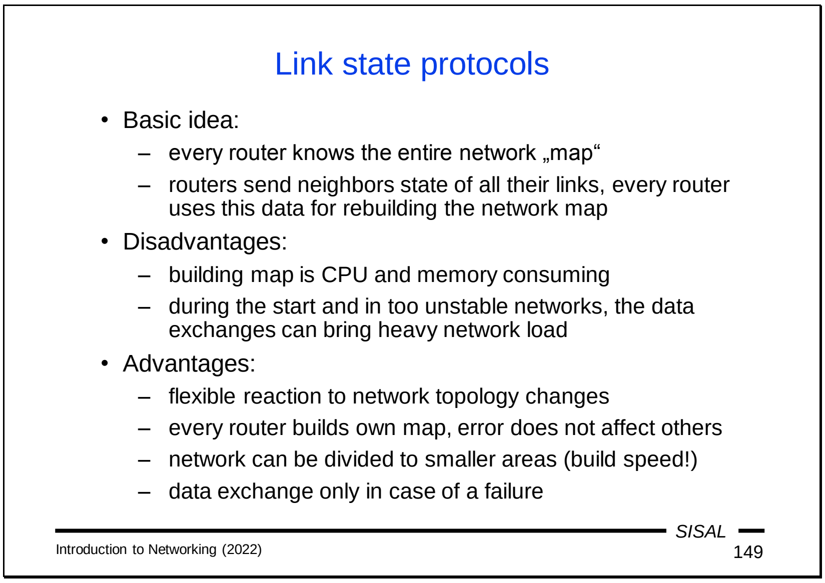
If the "infinity" were significantly longer, the service disruption would be significantly longer as well.

At the same time, this problem also illustrates one important principle in designing algorithms: there is always a need to investigate and prevent possible **race conditions**, i.e. situations where two independent actions can occur in an inappropriate order.

The negative effects of counting indefinitely are minimized in the RIPv2 extension:

- A *triggered update* is a mechanism by which a router sends an update **immediately upon** detection of a problem and does not wait for the update period to elapse. This will reduce the risk of a race condition that an update from a partner will arrive earlier. But beware, this risk will only be **reduced**, not **eliminated**, by this mechanism!
- In the *split horizon* mechanism, a router does not send a partner information about networks that it has learned about **from the same partner** (which would make no sense – the partner has better information about these networks than it does). This mechanism effectively **prevents** the race condition described above. A small tax on this is that the router cannot send the same update message to all its neighbors, but must prepare each one separately.
- A *poison reverse* is a slight improvement on the split horizon method – a router sends a neighbor data about the “neighbor’s" networks, but with a metric of 16, it "poisons" these records.

Slide 149



Link state protocols

- Basic idea:
  - every router knows the entire network „map“
  - routers send neighbors state of all their links, every router uses this data for rebuilding the network map
- Disadvantages:
  - building map is CPU and memory consuming
  - during the start and in too unstable networks, the data exchanges can bring heavy network load
- Advantages:
  - flexible reaction to network topology changes
  - every router builds own map, error does not affect others
  - network can be divided to smaller areas (build speed!)
  - data exchange only in case of a failure

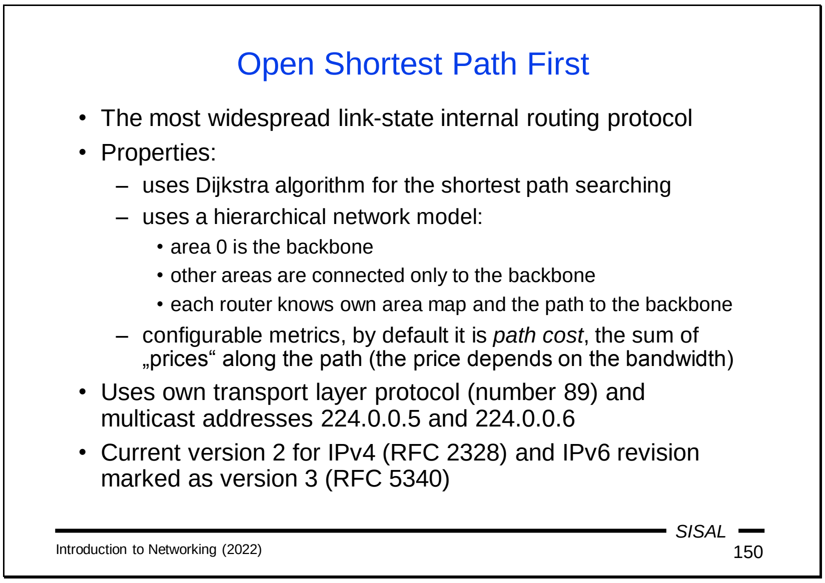
SISAL

Introduction to Networking (2022) 149

The second group of routing protocols are called **link-state** protocols. The name is based on the basic principle of these protocols – instead of a table with distances, only information about line states is sent. Each router holds the entire **network map** on its own and calculates optimal paths on its own, according to the line status messages it receives. If we use the car analogy again, in this case the driver has at his disposal not direction signs but a road map. If he learns about a problem on the radio, he can find the best alternative using the map on his own.

Even though this approach requires more computing power and (in the case of rapidly changing conditions) a certain network burden, benefits outweigh the disadvantages. Two of the benefits are substantial: the network can react far more **flexibly** to changes or outages, and each router makes its own calculations, so any mistakes **won't affect** any of its neighbors. On the contrary, the advantage for a stable network is that routers do not have to regularly exchange their entire databases.

Slide 150



Open Shortest Path First

- The most widespread link-state internal routing protocol
- Properties:
  - uses Dijkstra algorithm for the shortest path searching
  - uses a hierarchical network model:
    - area 0 is the backbone
    - other areas are connected only to the backbone
    - each router knows own area map and the path to the backbone
  - configurable metrics, by default it is *path cost*, the sum of „prices“ along the path (the price depends on the bandwidth)
- Uses own transport layer protocol (number 89) and multicast addresses 224.0.0.5 and 224.0.0.6
- Current version 2 for IPv4 (RFC 2328) and IPv6 revision marked as version 3 (RFC 5340)

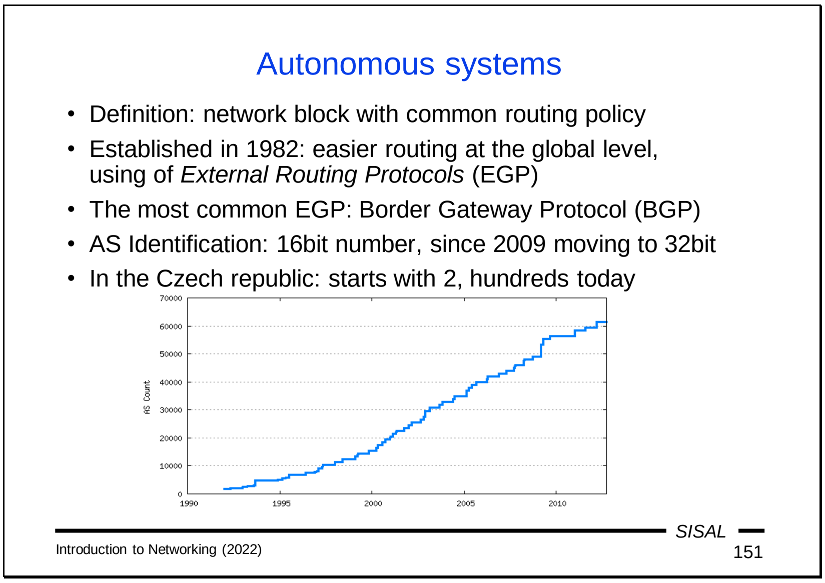
SISAL

Introduction to Networking (2022) 150

The most popular representative of link-state protocols is OSPF (Open Shortest Path First). Here, unlike RIP, **Dijkstra's** algorithm is actually used to calculate the network map. However, for large networks, even this algorithm would have problems with increasing network size and therefore an OSPF network can be divided into **areas**, so that each calculation is always done only within a single area and thus on a significantly smaller set of nodes. The network is arranged in a two-tiered schema: the area with number 0 is called the **backbone**, and all other areas are connected directly to it. Thus, any path in the network consists of no more than three parts: from the source area to the backbone, across the backbone to the target area's connection point, and across the target area.

The metric is called **path cost** in the OSPF, and is determined using a complex formula that is defined by the network administrator in a configuration. The path cost is able to include not only "distance" but also bandwidth, latency, throughput, and also the actual "price" of traffic along a line.

Slide 151



Autonomous systems

- Definition: network block with common routing policy
- Established in 1982: easier routing at the global level, using of *External Routing Protocols* (EGP)
- The most common EGP: Border Gateway Protocol (BGP)
- AS Identification: 16bit number, since 2009 moving to 32bit
- In the Czech republic: starts with 2, hundreds today

70000
60000
50000
40000
30000
20000
10000
0
AS Count
1990
1995
2000
2005
2010

SISAL

Introduction to Networking (2022)

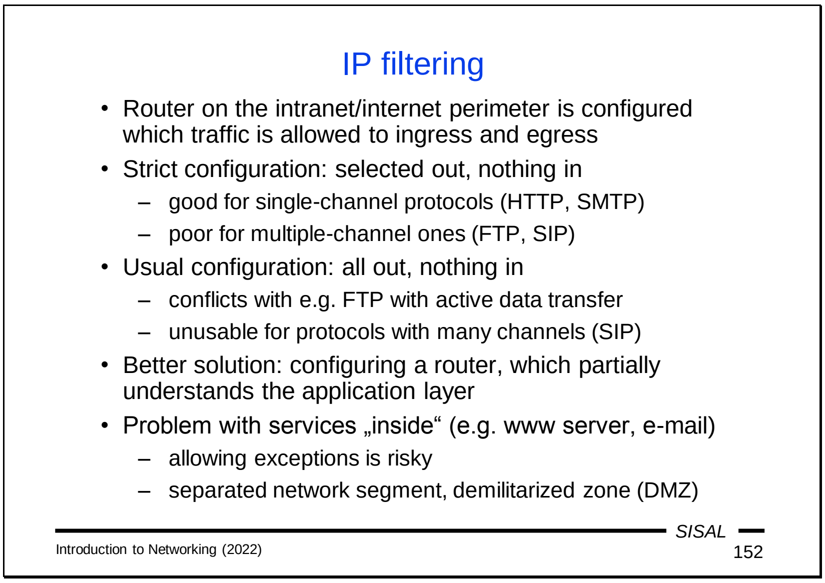
151

Until now, we've been talking about routing protocols in "relatively close" networks. But how does that change when we look at the Internet backbone?

Individual blocks of networks consist of so-called **autonomous systems** (AS). The exact definition is not very exact, it basically just means networks with a common routing policy, so it's not all that difficult to apply for an AS allocation, and so the quantity of AS numbers has increased rapidly. When the Czech Republic joined the Internet, it had two autonomous systems, and they now number in the hundreds. Typically, AS are owned by ISPs or large companies. In 2009, it was necessary to switch from the previous 16-bit AS numbers to 32-bit ones.

The purpose of autonomous systems is to unify routing for a whole group of networks at a global level. Routing between the different AS's then takes place on the basis of slightly different conditions than the one that takes place within each AS. It is not the network addresses that play a major role here, but the AS numbers. The protocols we have studied so far are called *internal routing protocols* (IGPs); so-called *external routing protocols* (EGPs) are used to manage routing between autonomous systems. The most common representative of an EGP is the **Border Gateway Protocol** (BGP). The critical features of an EGP are the inclusion of other factors in the evaluation of path metrics (similar to the calculation of path cost in the OSPF) and the avoidance of loops. Therefore these protocols use a **path-vector**, a sequence of AS numbers through which the path leads. Work with it prevents loops.

Slide 152



IP filtering

- Router on the intranet/internet perimeter is configured which traffic is allowed to ingress and egress
- Strict configuration: selected out, nothing in
  - good for single-channel protocols (HTTP, SMTP)
  - poor for multiple-channel ones (FTP, SIP)
- Usual configuration: all out, nothing in
  - conflicts with e.g. FTP with active data transfer
  - unusable for protocols with many channels (SIP)
- Better solution: configuring a router, which partially understands the application layer
- Problem with services „inside“ (e.g. www server, e-mail)
  - allowing exceptions is risky
  - separated network segment, demilitarized zone (DMZ)

Introduction to Networking (2022) *SISAL* 152

A router that connects a local network to the Internet is the point where the **security policy** of the entire network is applied. **IP filtering** is usually a primary part of such a policy. The name is somewhat misleading because it is not really filtering at the **IP layer**, but on the **transport layer**. It defines rules on what type of traffic (more specifically, traffic on "which ports") is allowed to and from the local network.

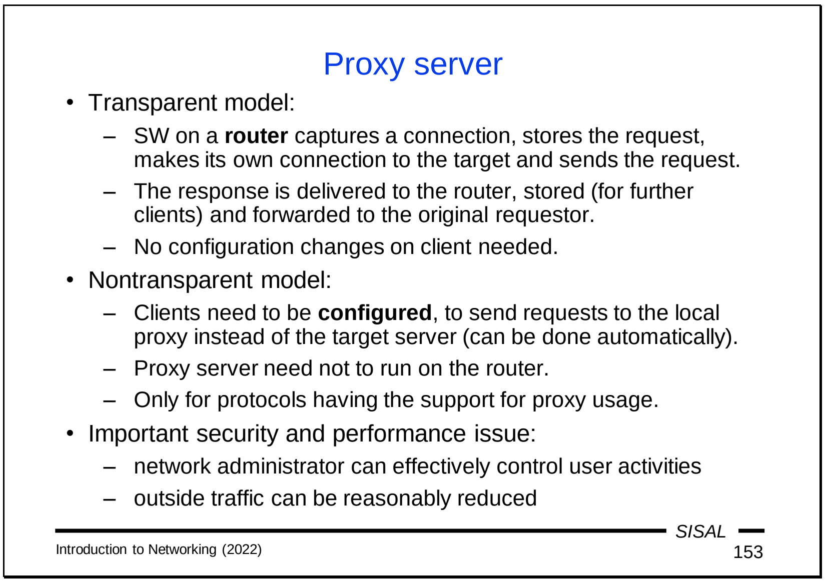
Note: The question is, what does “from" and “to" actually mean? In the case of TCP, of course, packets have to cross the border in both directions. What matters is who **opened** the connection (and therefore sent the SYN packet). In the case of UDP, we generally don't know the direction.

The most trivial configuration only allows traffic from the internal network to the Internet, and only on selected ports. This configuration is only acceptable for single-channel protocols such as HTTP or SMTP. When a protocol needs additional channels for its operation, the filter prevents opening them. The only option in this case is to use a filter that also understands that particular application protocol. For example, if filtering software intercepts an FTP communication in which a server and client are negotiating to open a data channel, it can “make a hole in the filter" for the two specific addresses, one inside and one outside the network, for a limited period of a few seconds so that the data link can be established.

A more typical configuration is not so restrictive regarding the list of ports that local clients are allowed to access. With such a configuration, for example, passively opening data channels in FTP will no longer be an issue. The issue will remain only with active channels or where more channels need to be opened (e.g. with SIP). Here, we can no longer get around without working with the application layer.

Another problem for filtering is providing services for the public Internet. In the past, this was a common problem, e.g. for a web server. Today, web servers are usually run by various **hosting** houses. However, if we want to run a server or other service on our own network, we will need to open a permanent hole in the filter allowing traffic from the external network to access a specific server and port. This obviously poses a risk, as once a potential attacker has an access into the internal network, he or she can exploit any software bug or server configuration error. Therefore, it is common to make a special segment of the network, a so-called *demilitarized zone* (DMZ), for such services. From the point of view of protecting the network, we consider it as a world that is neither completely safe nor completely unsafe. Therefore, filtering rules at the DMZ's borders with the internal network and the Internet are slightly more permissive than those at the border between the internal network and the Internet.

Slide 153



Proxy server

- Transparent model:
  - SW on a **router** captures a connection, stores the request, makes its own connection to the target and sends the request.
  - The response is delivered to the router, stored (for further clients) and forwarded to the original requestor.
  - No configuration changes on client needed.
- Nontransparent model:
  - Clients need to be **configured**, to send requests to the local proxy instead of the target server (can be done automatically).
  - Proxy server need not to run on the router.
  - Only for protocols having the support for proxy usage.
- Important security and performance issue:
  - network administrator can effectively control user activities
  - outside traffic can be reasonably reduced

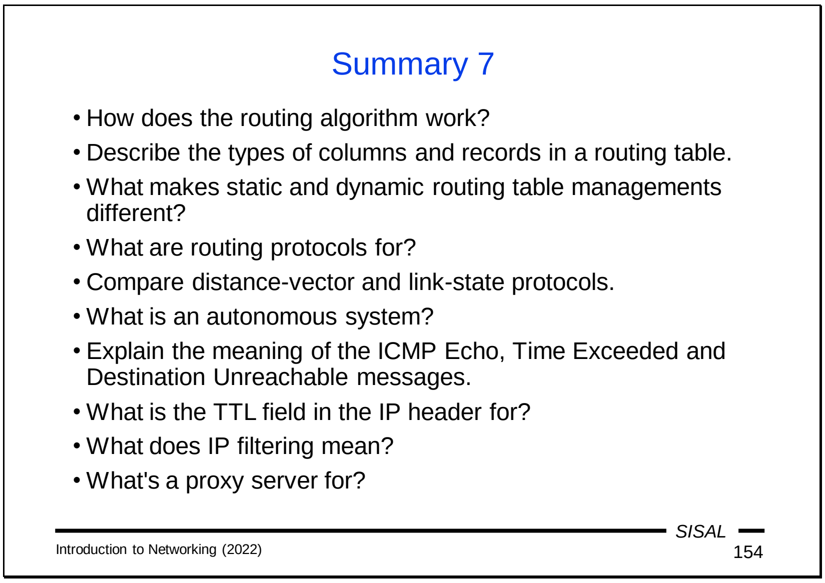
Software that controls the operation of a particular protocol (usually on a local network/Internet interface) is called a **proxy server**. It operates either transparently or non-transparently.

A *transparent* version of a proxy server works on a network's border router. The router intercepts a client's request and hands it over to the proxy server. The proxy server checks the request against the rules of the organization's security policy, and if the request is okay, it establishes a connection to the actual destination server as a client and sends the request to it. Once a response arrives, it is checked again (e.g. by an anti-virus program) and sent to the client. The advantage of a transparent proxy is that there is no need to change the configuration on the clients – each request reaches the proxy without the client having to know about it.

In the case of a *non-transparent* proxy, a client needs to know about the existence of the proxy because each request needs to be sent to the proxy server and not to the target server. A disadvantage of this solution is therefore that clients need to be properly configured (manually or automatically). An advantage is that the proxy server can run on a more suitable hardware or operating system than the router. But this solution is only available for those protocols that have support for it – the client must specify when sending the request that it is targeted to a proxy and tell the proxy the real target server address.

Using proxy servers has also security benefits (the proxy server can filter operations at the application protocol level) and performance benefits (the same request does not have to be sent repeatedly by the proxy server to the actual server; the response can be **cached** and sent to other clients by the proxy). However, the last point may gradually become less relevant for HTTP, as caching cannot be used together with HTTPS.

Slide 154



Summary 7

- How does the routing algorithm work?
- Describe the types of columns and records in a routing table.
- What makes static and dynamic routing table managements different?
- What are routing protocols for?
- Compare distance-vector and link-state protocols.
- What is an autonomous system?
- Explain the meaning of the ICMP Echo, Time Exceeded and Destination Unreachable messages.
- What is the TTL field in the IP header for?
- What does IP filtering mean?
- What's a proxy server for?