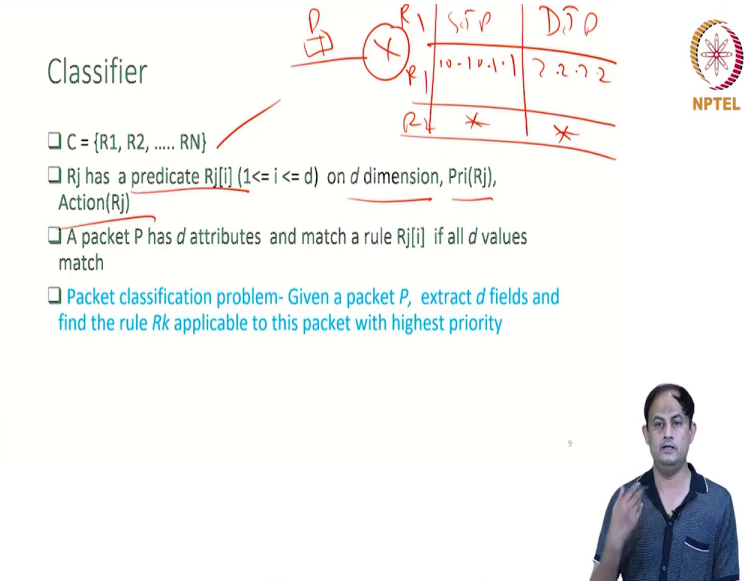
Advanced Computer Networks
Professor Dr. Neminath Hubballi
Department of Computer Science Engineering
Indian Institute of Technology, Indore

Lecture 10
Packet Classification – Part 2

(Refer Slide Time: 0:18)



So, before we actually look into the algorithms, let us formalize what the classifier is. Then we will look into some of the data structures and the mechanism to do this, differentiate its service or the packet classification. So, we said that the classifier is denoted with C, and it is a collection of the rules. Here, in this case, there are n rules R1, R2, and RN, and every rule has got few things, one is called the predicate, Rj is the rule, and maybe Rj is the 10th rule.

And this predicate is on every dimension, i.e., the source IP address is one dimension, the destination IP address is another dimension, and you have a condition to evaluate on every dimension; some of the dimensions might have a wild card like * that is okay, you have got a condition on every dimension, * is also a condition, it is matching to every possibility that is the difference, it has got a condition on the d dimensions.

And then the second thing that it has got is a priority assigned to this particular rule. We will come to that in a minute, why this priority is required, and then it has got certain action to be performed. So, a predicate is a combination of the predicates that are there in the rule and then an

action to be taken, and then as the third one you have got is a priority assigned to that particular rule, these three things are there.

Let us say that a packet P is arriving at a router R1, we pick up the header portion of that packet and exactly d attributes from that particular packet and we say that a packet P actually matches a rule Rj if all the d values d1 to d2, d2 to whatever dk, you have k number of the dimensions, if all the k attributes match then we say that this rule is actually matching.

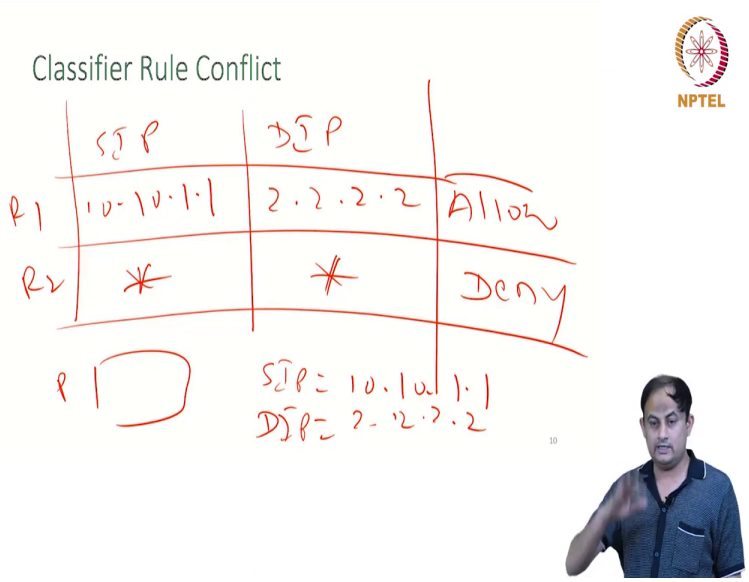
The packet classification is all about, given a packet P extract d fields do matching and if all the attributes that are required match in that particular rule for this packet, then we say that the packet is matched and it is not necessary that only one rule is matched, it is possible that more than one rule can match. So, that is why we need to pick up a higher priority rule if there is more than one match, then you need to pick up the rule which is having the highest priority.

Just an example, if the attributes are, let us say, source IP address and the destination IP address and one rule says 10.10.1.1 is the source IP address and the destination IP address is 2.2.2.2, this is rule number 1, let me call it R1 and the second rule R2 says that this is *. So, it does not matter what the source IP address is and what the destination IP address is. So, it is easy to note that R1 is actually a subset of R2.

So, R2 matches anything no matter what are the source and the destination IP addresses, any packet that is coming to the router will always find a match with the R2 but R1 is a subset of that. Now the question is if, let us say, R1 says you need to forward it with priority, R2 just says that you need to forward it, whether I forward it with the priority, or whether I forward it by just following a normal priority that is the differentiation we need to bring and that is actually resolved using the priority value.

So, that is what the packet classification problem is you pick up the packet that has arrived, you pick up the header values in that one and you got a set of rules inside your classifier, evaluate against the rules inside your classifier, and come back with the highest priority rule that is matching with this particular packet header fields that is the classification problem. Now any algorithm that I want to design needs to do this kind of operation.

(Refer Slide Time: 5:16)



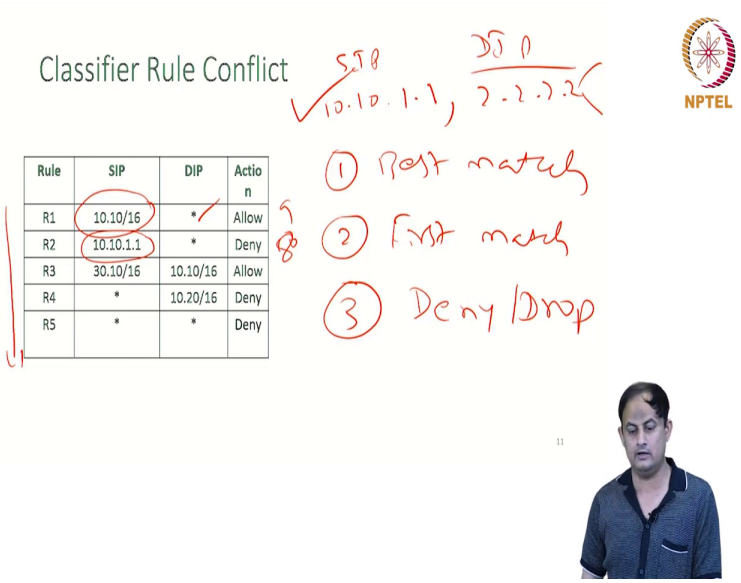
But the example just I gave you led to something called the rule conflict. So, the previous example that I took again on these two fields just to simplify the discussion the source IP address and the destination IP address are the two fields; one is 10.10.1.1 going to this 2.2.2.2, one is saying that you allow this communication and the second one is saying irrespective of what is the source and destination IP address, you actually deny this communication.

So, let us say a packet P comes, and the P has got the source IP address of 10.10.1.1 and the destination IP address of 2.2.2.2. Now when I evaluate against this set of rules, both R1 and R2 match, and according to match whether I am going to allow it or deny it, we said that we need to assign a priority to this rule, if R2 has got the highest priority then you need to deny this particular packet to go through, if R1 has got the highest priority then you should allow this particular packet to go through. The more fields I got, the higher the chance of this conflict.

So, more than one rule might match because, for the same combination source destination IP address, if I add a protocol field, one with the TCP field and another with the UDP combination, I can possibly have two different rules inside my classifier. So, now as the combinations increase, the number of rules will also increase; the more the number of rules you have got inside your classifier more the chances of a conflict.

Now in order to assign priority to these rules, I need to understand the priority and how do I assign priority to these rules. So, that brings us to the discussion to something called conflict resolution.

(Refer Slide Time: 7:28)



So, let us try to understand that rule conflicts are very much possible inside the classifier, one is telling something else, and one is telling something else that is possible. So, now how do I actually resolve the conflicts inside the rules when I say conflict resolution, basically, I want to assign the priorities in such a manner that conflict resolution is always possible. So, meaning, can two different rules inside my classifier have the same priority, maybe priority number one is assigned to more than one rule; if that is possible, again, there might be conflicts; how we actually resolve these conflicts is the issue.

So, let us try to understand the strategies or different mechanisms used to resolve this conflict. So, there are five rules in this particular classifier and the first one is saying 10.10/16 this is a class 16, which falls in this range irrespective what is the destination IP address you should allow this particular packet to go through.

So, again let us take the case that a particular packet comes with this combination 10.10.1.1 this is the source IP address, and the destination IP address is let us say again, 2.2.2.2 anyway, the rule R1 and R2 are the candidates now based on the source IP address and the destination IP

address combination for both the rules are * this is anyway going to match, R1 is going to match, for this is the best case.

And now you look into the other case now R2 has got the source IP address, you see this is not the prefix format; this is the entire IP address. The other rule R1 is in the prefix format, now which rule to apply or which rule to prioritize, maybe I am going to say that this has got the highest priority, maybe the higher the number higher the priority, you can have the other way around as well and this has got the priority number 9.

So, now the highest priority rule for this combination is R2, and based on whatever it says, it says deny, you actually deny the particular packet to go through. Now the question is, is this the only way with which you assign the priority, rule with the highest priority you apply that, that is one way of resolving the conflict,

Then there is what we call the best match that you have got. So, one is in the prefix format; there are many, many other combinations in this prefix format 10.10/16 another 2^{16} combinations fall under this series that is not the most specific one; rule R2 is telling the exact source IP address, so that is the best match that you can find. So, now if you use this strategy, what it means is my classifier, in order to do a packet classification or a decision on that particular packet, you need to start with the rule R1 and then go across the table and evaluate all possible applicable rules, find out that and then you actually come back with the one which is actually the best matching one. So, there could be many, many other possibilities; you can have another rule saying 10.10.10/24 in the prefix format; now, there will be not only two, there will be three rules which are matching to this particular combination.

So, among all of them, you need to find out the best case; the second way to resolve this conflict is something called the first match. What it means is there are some rules inside my classifier and I am going one after the other, rule R1 R2 R3; you first apply the rule applicable, which is the first one in the sequence, and you take the decision on that one, in this case, it also happens that if you get this particular packet rule R1 is the first match using that it is saying allow, you actually allow this.

Now if you follow the first match category, what it requires is I want R2 because R2 is the more specific rule applicable to this particular combination, R2 to come ahead of R1. So, maybe the

order of these rules needs to be interchanged. So that I can make the decision of routing or forwarding based on the longest prefix. So, the same kind of rule can also be applied here as well. So, R2 is more specific I want that to apply I want to use the first match category by default the rules inside the classifier need to be ordered in the decreasing order of their priority. So, which is having 10, R2 has the highest priority that should come first and then should come the R1 that is another strategy that you can use.

And the third strategy that you want to use is if some rule is saying you need to deny or drop the packet, you go with this one because denying or dropping is more conservative in nature, particularly if you are implementing a firewall kind of operation some two systems you do not want to communicate.

So, in this case, rule R2 says that has got the best match that you can find, although this has got the lower priority; let us say the priority of this rule is 8 and the priority of rule R1 is 9, and the highest priority rule is the R1 but because R2 says you need to drop this particular packet and dropping is more conservative in nature and it is best for the security. So, we want to apply that rule, whichever is saying you drop the packet.

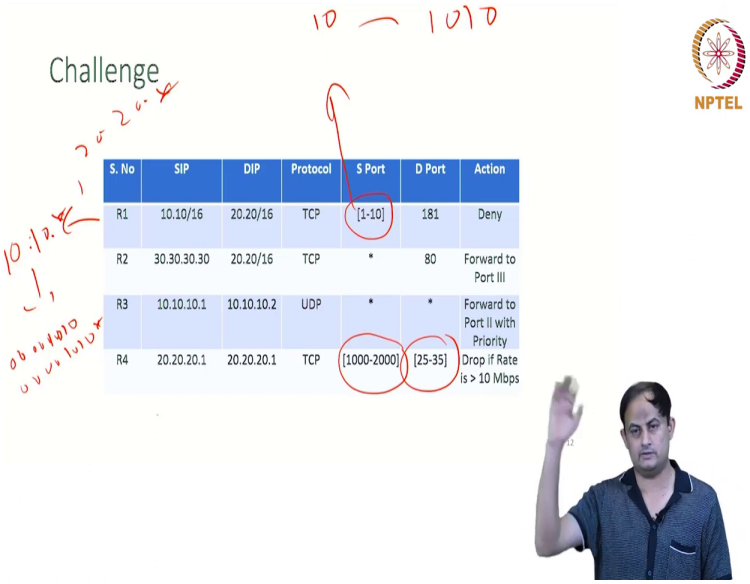
So, I am going to again look into the classifier all the rules and then find out all the applicable rules and again start using the priority. Among them, if some rule is actually saying that you need to deny this particular packet, you apply that rule to that particular packet, this is how the evaluation is done. One is the best match that requires if your rules are not ordered, you need to traverse the entire set and then you come back with the best match that is possible with the highest priority.

The second one is if I have the ordered rules inside my classifier, find out the first match, whatever the rule is applicable, then you apply the action that is specified by that rule and the third one are you give priority to that rule which is actually denying the content. So, this is actually facilitating this kind of operation is not an easy job because you need to have may have thousands and thousands of rules. I can constantly keep inserting new rules inside my classifier.

Then if I insert one rule that might conflict with so many other rules, I need to reorder the rules inside my classifier; what is the best order, so many things can get disturbed or if an existing rule is taken out from this one, again you need to re-alter the table and all this can happen. Now we

will try to understand how exactly this kind of rule matching and the applicable action can be taken on that particular packet when you want to do the differentiated service.

(Refer Slide Time: 15:38)



So, another challenge is whatever the rules that we saw in the previous case did not have the rules specified in this particular format; either there is a wild card or there is a number, but that might may not be always the case. So, what is the challenge here is sometimes the rules might not necessarily have the exact numbers but a particular range for example, rule R1 has got this source port in the range 1 to 10, and rule R4 similarly has got a range of 1000 to 2000 in the source port field and similarly for the destination port 25 to 35.

Now, how do we actually evaluate? it is not only one port if I have to do a matching, I need to check whether the source port number in that particular packet falls in this particular bracket or not. Now if I want to do a quick lookup or the matching of these rules this is similar to that of the IP lookup table, and what we understood in the IP lookup table is if the rules or the dimensions can be expressed in the form of a prefix. So, in this case, this rule for the source IP address can be expressed as 10.10.* does not matter, what is the next combination that is the source IP address and then 20.20.*. So, this is in the prefix notation I can always convert it into binary maybe if I convert this into binary, you will get 0001010, which is the first eight-bit and then the second eight bit would be 0001010, and then you put the *, that is the prefix for this particular source IP address field in the rule R1.

Similarly, for R2 and R3 and so forth, and similarly for the destination IP field also, you can have this conversion. So, the source and destination IP addresses can be expressed in the form of prefixes but not the source port number and destination port number, which are expressed in the form of the series in this range. How do I convert the port number, and sequence of port numbers that are expressed here from 1 to 10 into a prefix format?

Remember that if prefix format is an exact IP address or the exact number, that can be a prefix format. For example, if the source port number is 10, what is the prefix notation for this in binary 1010 is the notation. So, I can say that the prefix notation is 1010; this is the exact match this is fine.

So, now it is not only one there are ten different IP addresses in this IP address field; how do I actually convert this into prefix format, remember the data structure that we studied may be the trie one, or the hardware implementation that we studied for the route lookup, are all applicable or efficient when the fields or the constraints or the predicates are expressed in the form of the prefix notation.

So, the one way to do this is take rule R1 and the source port is actually saying expressing a range and for every possible source port number, you write a rule, i.e., rule R1 is broken up and for every source port value, there will be one rule inserted into this classifier. So, the same source IP address, destination IP address and protocol field when port number is 1, you want to set the rule one. When the same source destination IP address and the protocol field when the port number is 2, you have a second rule like this one now. Because there are ten source possible port numbers, there will be 10 different rules written inside the classifier.

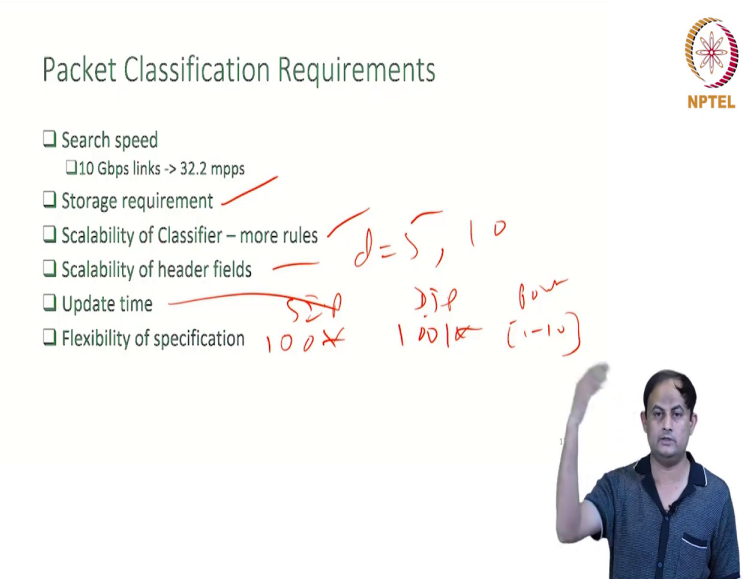
And it is no secret that if I do that then the number of the rules inside the classifier are going to increase, and they will be blown up. Now the more number of the rules you need you to have in the classifier, the more work you need to do because I do not know whether R1 will match R2 will match, n number of rules are there and you need to evaluate all of them, so that is actually not desirable.

Now the question is, is there another way of doing this, can I express the port numbers written inside this range format into a prefix format, there is a way to do that; we will come back and

learn that part later. So, assuming that is a challenge, the reason why I am talking about this right now is the set of the data structure that we are going to study is efficient for the prefix notation.

So, if I use the prefix notation to express all particular combinations of the values inside my rule, then I can implement the evaluation part quite efficiently; with that background, we will come back and see how to convert this range into a prefix format sometime later, but assuming it is there then we will try to do that.

(Refer Slide Time: 21:16)



Now assume I have done all this, all my constraints are specified in terms of the prefix notation. Now, what are the practical requirements, how do I evaluate the performance of one particular kind of implementation, and what are the advantages of a particular implementation? One is the search speed, how much time it takes because, of course, when you want to do a matching over a set of rules how much time it takes, more the number of rules you have got, the more time it might take, the efficiency or the speed with which you want to do this matching will happen.

So, if you have a 10 Gbps link, then approximately 32.2 million packets per second, that is the speed with which I want to do the classification; there is a practical requirement. And the then the second thing is, how much storage do I require to do this kind of classification, if I come up with the data structure and if it is taking too much amount of space, then that is not desirable.

So, the efficiency of the storage of that data structure itself becomes an important parameter for making the decision, and the data structure or the algorithm that I use can accommodate more number of the rules. Today I have 100 rules; tomorrow, I might have another 100 rules, and the sometime down the line, I might have 1000s of rules.

So, can the data structure scale well to accommodate new rules to be inserted in the table, how much change that I need to bring inside whether it requires the complete rearrangement or rebuilding from scratch, or can I build incrementally alter the data structure and accommodate new rules whether that flexibility is there that is another requirement, it needs to be flexible to accommodate that.

And today, I got d number of fields; maybe d is equal to 5 fields today; tomorrow my dimension might increase to 10. Can the data structure accommodate that increased number of header fields? And then the fifth parameter is how much time it takes to do the update operation, whether I can do it quickly or maybe a fraction of a millisecond or it takes some time, whether there is downtime involved if I do the update, take out the old set of the rules and put the new set of the rules inside my router, if this requires the downtime of the router that is not desirable.

And the last parameter is the flexibility of the specification. If you go back to the previous example, the source IP address and destination IP address can be expressed in the form of the prefix. This is the source IP address and this is the destination IP address but the port number needs to be expressed in the form of the range; as a user, I specify this range, then the router automatically takes this and then does whatever format it wants to convert it into automatically and then do it.

So, that gives the ease of specification to the end user if I am the network administrator, my job becomes easy. So, I can write a minimal set of rules, and then whether it is replicating, creating a multiple numbers of rules out of one, or converting the range into prefix format, whatever it is doing related to the router it will automatically do.

So, the kind of flexibility the router provides, the better it is. So, whether we want to, whether all of these can be met or not is a question. Still, at least in theory, these are the kind of the parameters we want to take into consideration when we want to implement or make a choice of the one kind of data structure over the other one or one kind of the algorithm over the other one.