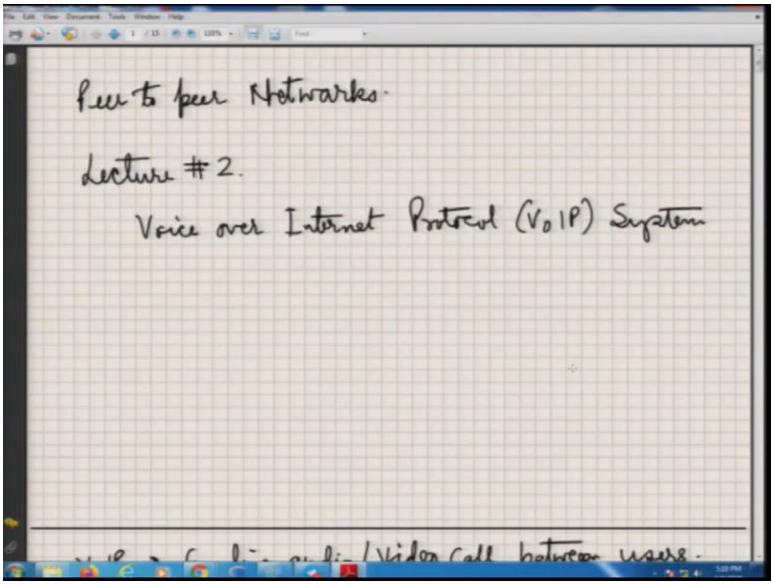
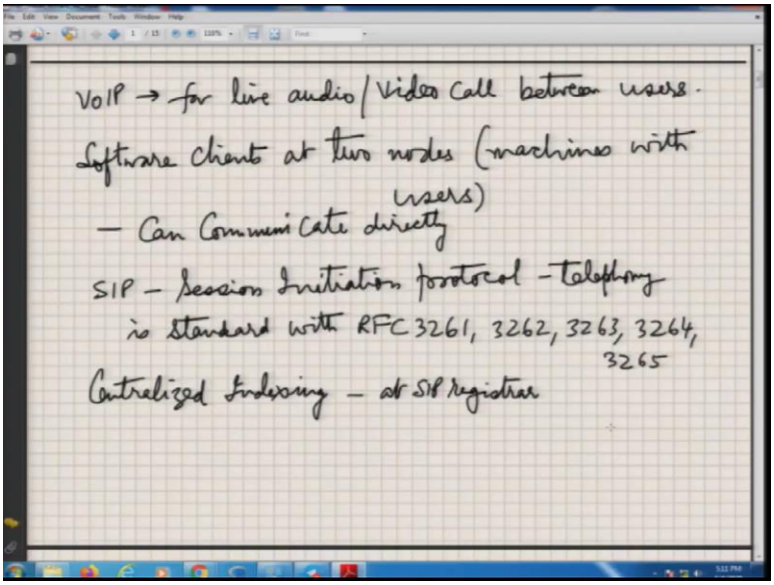
Peer to Peer Networks
Professor Y. N. Singh
Department of Electrical Engineering
Indian Institute of Technology, Kanpur
Lecture 2
Peer to Peer Networks in Telephony: Voice over Internet Telephony (VoIP) and Distributed Hash Table (DHT)

(Refer Slide Time: 0:23)



Welcome to lecture number 2 of this course. In the previous lecture, we had talked about some general principles, such as how the security or authenticity of a website is being ensured and how the authenticity of two people can be done mutually between them without any third party. And I have also talked about how the certificate gets signed by a certification authority. So we will now move ahead and then try to look into how the voice over Internet protocol, voice over IP system in telephony works. This is a good example of how the peer to peer network is being used in telephony. So in voice over IP telephony, nowadays, we use something called session initiation protocol, SIP.

(Refer Slide Time: 1:11)



This SIP protocol is the kind of now default structure by which all telephony systems are built. And this is an example, a standard example of a voice over IP. So we still use now IP Internet protocol-based packet switching network, on top of which the voice is being transmitted. You cannot only do audio calls the way it used to be done in earlier days with telephones, but you can also now do video calls.

And the way it happens is generally in such kind of communication two endpoints or two telephones; the voice is not going to an exchange and then from there exchange not over to the lines to multiple exchanges and then to the other side. So either endpoints or both telephones will be connecting to the Internet. And somehow, the connection will be set up between these two endpoints, and the communication will be happening directly between them.

So whatever I am going to speak on a telephone will be converted to packets, IP packets, and those IP packets will have a destination address for the other endpoint. And the other endpoint will receive this; all these voice packets will be played back for him. And similarly, in the reverse direction, the voice packets or voice information will be packetized and sent back to me, and we both will be talking. So exchange will not know what is happening; the exchange role is only to set up the call. So this also we call data call. And that is how we work nowadays.

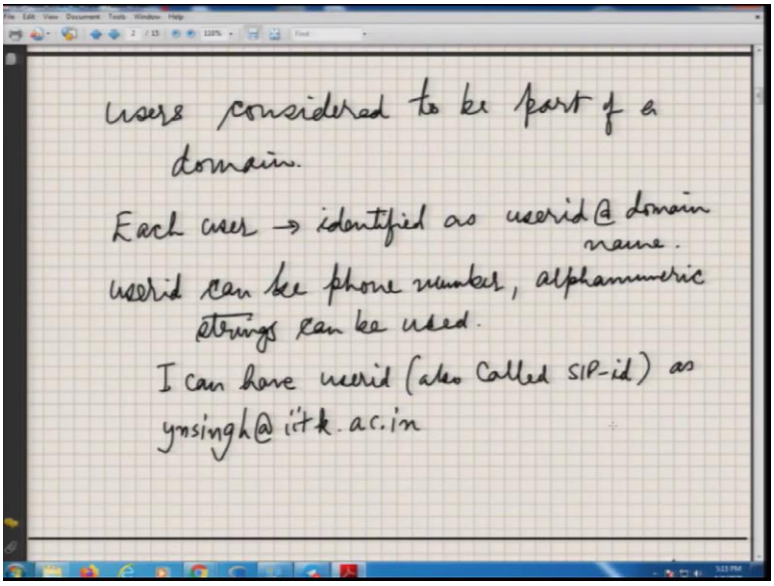
So, this session initiation protocol is also known as SIP, SIP-telephony. And now this, there are standards which have to be followed to implement such a telephony system. And these standards

are coming from ietf.org, Internet engineering task force, so there the standards are built. These are all open standards built through an open process. But kind of now become a defacto standard for everybody to follow.

And these follow the RFCs 3261, 3262, 3263, 3264 and 3265. And there are more variations which have come, so you can go to the website and search for these RFCs and go through them. And there is a lot of learning, which one gets when you go through an RFC and try to understand or gain insight into it. This particular thing comes in the peer-to-peer network category because two endpoints are directly talking to each other.

But, how you identify that what is IP address and port number of a certain user? How will that be identified? Normally, the user ID will be a phone number in this case, and there will be a corresponding IP address and the port number on which communication can occur. So all this information will always be stored; there is something called a SIP registrar. So the indexing here is centralized. It is not distributed indexing. We will come to distributed indexing as I go, goes along in the latter part of this same lecture.

(Refer Slide Time: 4:20)



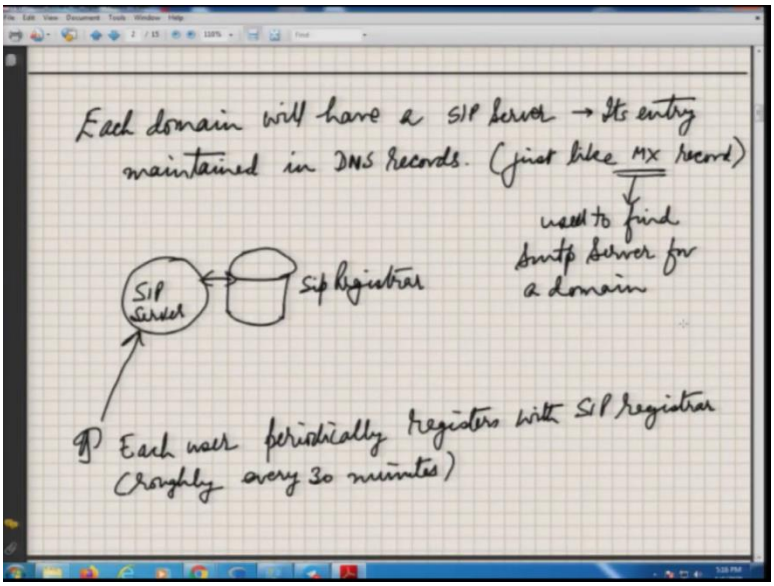
Normally, what happens is, users are being partitioned into domains. So, for example, when I am in IIT Kanpur, IIT Kanpur is a domain. We follow the same system for emails. So when you send an email to ynsingh@iitk.ac.in, the domain is iitk.ac.in. So IITK is a sub-domain of the ac

domain; ac is academic. Academic is a sub-domain for India. So .in is one full domain which is representing a country.

So the way we have email IDs, I will have a phone number@iitk.ac.in as my user identification. These need not be phone numbers, though, because we used to use that numbers in earlier days, so still, I think there is a tendency to use the phone number@iitk.ac.in. The telephone exchange which we have in IIT Kanpur is a SIP-based exchange. For example, a 4-digit phone number that I have 7944 actually should be written as 7944@iitk.ac.in.

But for the whole IIT, there is only one exchange. And so every user in IIT belongs to the same domain. So we do not explicitly write @iitk.ac.in. We just write four digits; the rest is automatically assumed to be, assumed by the exchange that the other end is also, this user belongs to the same domain for which this is the SIP, server or SIP registrar. So this domain-based organization is also used in this case.

(Refer Slide Time: 6:03)



Now, normally each one of this domain will have a domain name server, DNS. So, when you send an email, if you want to send a mail to www.gmail.com. For example, I have an email ID at ynsingh69@gmail.com, and if you want to send mail there, you have to find out which is the corresponding mail server in gmail.com. So for which, you will go for the DNS server, which is responsible for gmail.com. Now, who will maintain this record? So there is going to be a DNS

server for .com, which will maintain the record that what is IP address and port number for gmail.com.

Similarly, .com, whatever is the DNS server that will be maintained by something called root authority or dot, which is being. They are well-known root authorities. You can always go to root authority and find out which is the DNS for .in. And then from there .in, you can find out which is the DNS for ac.in. And keep on doing it, and you can find out what is a DNS for iitk.ac.in. And when you go to iitk.ac.in, you will find out; you will inquire about something called mail exchange record, MX record, which is maintained.

And this will give you the SMTP server of IIT Kanpur. You can then directly send all the mails for iitk.ac.in domain to this particular SMTP server. That is how all mail servers from all across the world directly communicate to our mail server by identifying what is the corresponding IP address through reverse chaining. And they also do a reverse zone mapping. They will find out, given the IP address, what is the corresponding DNS name.

So they both have to be done. So that is the kind of check and balance which we keep. And then, all the mails are directly delivered to our SMTP server. And all IITK users will pick up the mail from this SMTP server. So the SMTP server dumps it in a mail storage that is then accessible to people through IMAP, Internet mail access protocol.

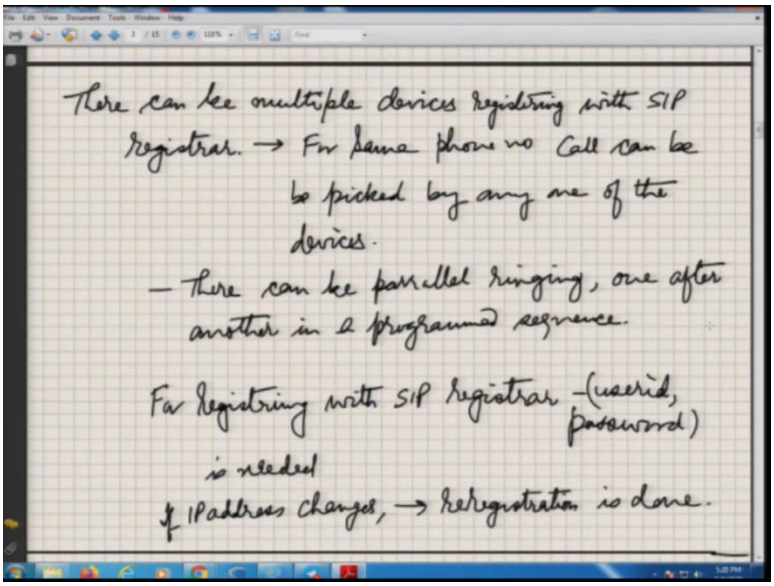
Now, something similar like this, we will also have a SIP record and SIP server record in the DNS. When I want to call somebody, I will always hand over my request to call my SIP server in my domain in IIT Kanpur. And IIT Kanpur SIP server will then find out if a call has to go to some phone number@iitb.ac.in; it will search for the DNS record or DNS server iitb.ac.in. We will ask him what the corresponding SIP server IP address is and port number for him. And then, my SIP server will talk to him, and then we will try to push on the call. An IITB server can identify me, our SMTP, our SIP server in IITK corresponding. That is how, essentially, the whole structure has been designed.

So another important thing that happens is my IP address can change. I may be on the Wi-Fi; I may be moving around. So if some call comes to me, it should exactly come to me only, not to somebody else. So my SIP server will also have a database. We call it the SIP registrar. So it will now maintain the entries of all phone numbers and a corresponding IP address and port number

continuously. And normally, this entry will purge automatically after 30 minutes. It should be refreshed, or again it will be republished by the client, even before this 30-minute time limit gets expired. So this is done periodically.

For example, every phone, my office phone, which has a number 7944, will periodically register with the SIP server and will tell, this is my IP address, and this is my port number. So this is what is going to be informed. So at any point in time, if I move my phone from my office to some other place, a different IP address will be assigned to it through a dynamic host configuration protocol, and that will get registered and overwritten in the database. So that is how the indexes are being maintained, and these indexes are kind of pointers to find out where the other user is and how I can talk to him.

(Refer Slide Time: 10:29)



So, we can even go to an extent where my office phone number may not be only on my office phone, but I have, probably have a lab, I have another office, I might put the same thing in my SIP line on my mobile phone. But all of them will have the same phone number, 7944. So there can be multiple devices registered with the SIP registrar for the same phone number.

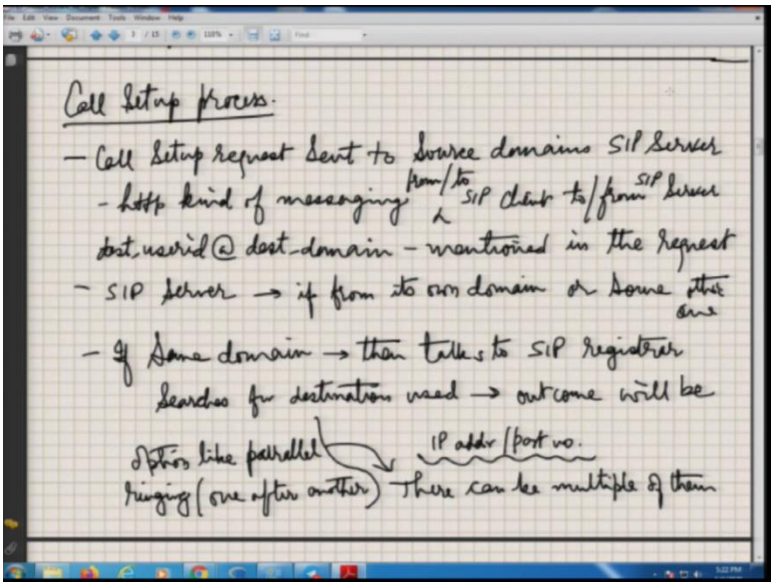
And I can pick up the call from any one of them. Whenever a call comes, all of them will be ringing together. And this can be a parallel ringing, or there can be sequential. So, first of all, it goes to my office phone. I do not pick up for within 5 seconds; then, it will start ringing in my lab. I still do not do it within 5 minutes, five seconds, then maybe my other office phone will

start ringing. If I even do not pick up that, my mobile phone will start ringing—the client is connected through Wi-Fi.

So this client is an Android client. We do get, CSIP is one available client, which does this job. And but all these devices have to register with the SIP registrar. So this will be all using my phone number and password. All phone numbers have been given a password. These passwords are known to telephone people, and the telephone exchange unit knows that. So they configure it.

So I knew for mine because I was handling that exchange for some time as ahead. So I know what my password was then, so I could have configured my office phone on my laptop also, so I did that. And an IP address changes automatically means, again, re-registration will be happening and overwriting will be happening. So multiple devices, of course, have to be permitted at the registrar.

(Refer Slide Time: 12:17)

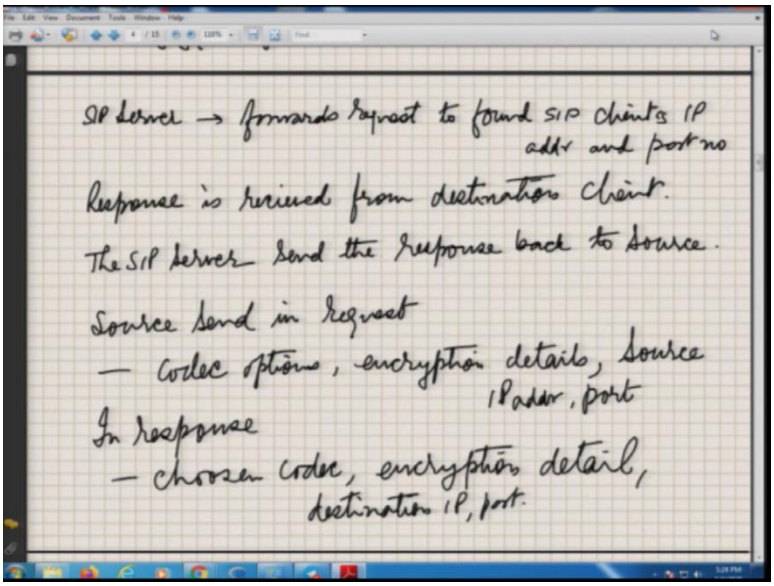


Now, what will be the call setup process? So call setup request, this almost like an HTTP, hypertext transfer protocol kind of thing. It is a similar format request and response. In this case, the source domain SIP server means the IITK SIP server will send all my requests to it. And it will be messaging, basically a simple message, a text message will go. And it will be from my SIP client to my SIP server, and I will, the message will contain destination user ID, so maybe a phone number and a destination will be going in the request.

Now SIP server will figure out if the call has to be routed from the current domain to within the same domain. If it is in the same domain, life is very simple. That is what happens when we make internal phone calls within IIT. It will simply talk to the SIP registrar, a database server, as I mentioned earlier. And it will search for; this is the destination number, what is the current IP address and port number for this.

If there are parallel things registered, I need to get the IP addresses and port numbers of all endpoints. And we will send, a message will be sent to all of them if it is a parallel ringing. And I will wait for the response from them. So ringing will be generated locally by these devices if they are on. It is not that ringing current is being sent by the SIP server. No, it will not be. It will locally generate it by the device.

(Refer Slide Time: 13:44)



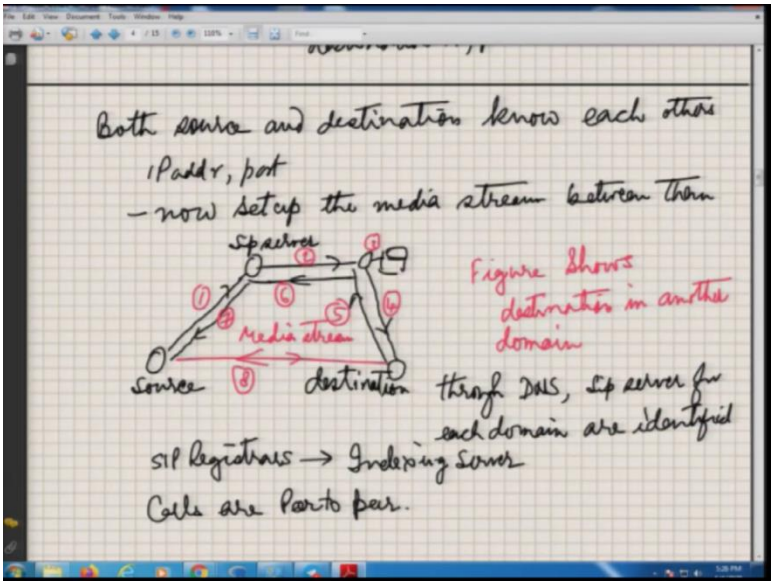
Now SIP server will forward the request to all the SIP clients, which have been found with the registrar, so all IP address port numbers will be known. Now, the response will be received from any one of these destination clients. This response, now normally in the request, will also have this kind of codec I will use. This will be my handshake for the secret key; the security has to be done. So all those possibilities, my IP address port number for media or voice call will be this. All this information will be going from source to destination.

Destination will respond, say that I can only use out of all the options given by you; only this particular code X, my IP address port number is so and so, my handshake power component for

the security key is this. So this will be going back to the source. So the source will have both the parts; it can create a security key. It will know what the other endpoint address is. It will know which codec is being agreed upon, so they will essentially agree on the same codec and start a media call.

So codec interruption, destination IP port, source IP address port will be known to both the entities, and a media call will be set up, and people will talk. But the problem is in earlier days, when you are making a telephone call, somebody at the telephone exchange can always listen to your call. So call can be tapped. And of course, it is a legal requirement in most countries that the call should be tapped for security reasons when rogue people talk actually. So for ensuring law and order, this is a needed requirement. This is by law. It usually is not being invoked. But of course, every good thing also has a bad aspect. Some rogue governments that are oppressive also use this to essentially find out who the corresponding people are not in agreement with the government to take care of them. So it is also used for that. So it has a double-edged sword, which happens both ways.

(Refer Slide Time: 15:46)

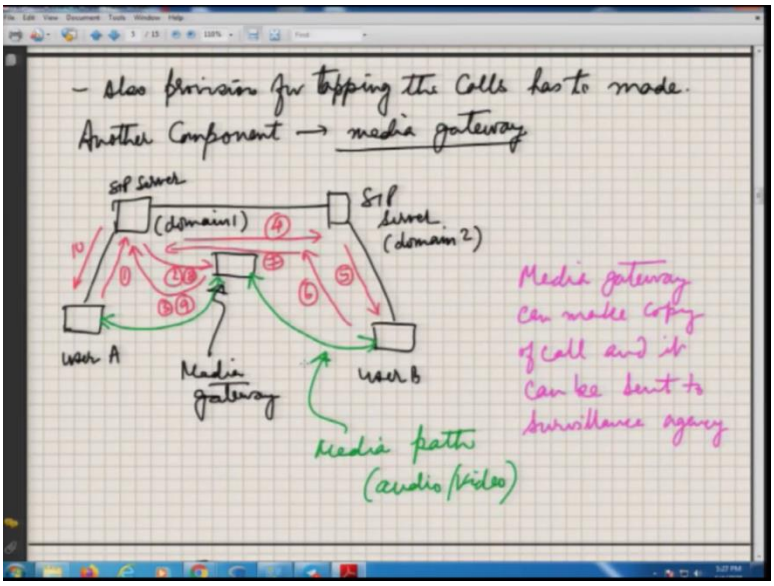


So this is being shown in the figure. How does it happen? So, at step number one, the source is sending a message to the SIP server. A SIP server is sending the message to the other SIP server. Another SIP server is the guy in a different domain now, and he is looking at the registrar to find out the corresponding IP address port number. The fourth number, the request, passes on to him.

Fifth is whatever he is agreed upon, which will be sent back as a response, which again goes back to the source SIP server, and back to the source.

Now, these two guys know each other's IP address port numbers, the session keys and set up a media call between them. This is typically the process which goes, and there is a peer to peer connection. Now you can see, there is no server in between. Both are equal entities; both are peer clients.

(Refer Slide Time: 16:42)



As I said, sometimes tapping has to be done; it is legally required in almost all countries. So we need to keep a provision on how to make a copy of the call and route it to the law enforcement agencies. So for this, we need something called the media gateway. So, in this case, this particular figure shows this aspect. First of all, the request will go to the SIP server. This is step one.

SIP server will now talk to the media gateway and will inform him that what is the corresponding port for which the source should be connected. It will keep it with itself. It will not still tell. It will also find out which source ID and port number user B need to be connected, or the destination to be connected. This will be coming, and this information will be coming back to 3. Then, of course, the 3, still this media gateway, also needs to be told the corresponding IP address. This will happen in the 8th and 9th steps.

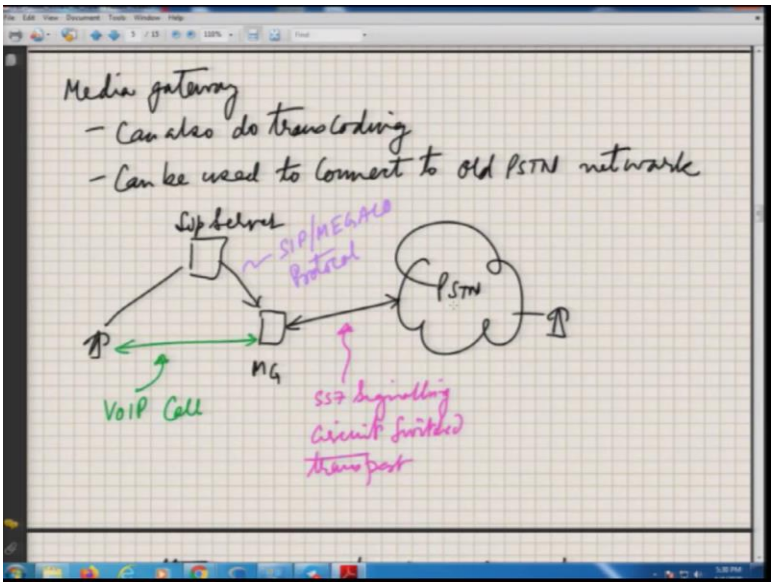
So the SIP server will now send the request to the SIP server in domain 2, which in turn will send the request to user B. User B will now respond back, will also tell what IP address and port number, in this, we should go via 6 to the SIP server of domain 2, then to SIP server of domain 1, which in turn has to report back to this media gateway, what is the corresponding IP address port number of user B. So it gives confirmation. Now it will go back.

Remember, when this 4th request went, that time SIP server of domain two has told user B, the source's IP address at port number is the same as the media gateway's IP address and port number call has to be plugged in. The call has to be connected. Now then the reverse for the user A or the source, this IP address and port number will be communicated back for the communication. And in turn, the SIP server is, which is primarily responsible for this media gateway, will talk to him.

Will set up the calls, the addition of these calls and other configuration, and route it to some other person who is probably a law enforcement agency, tapping the call. So encryption key will be between this and this, media gateway and user B and between user A and media gateway. So then this, this call essentially is being kind of a plumbed together. It is plumbing work that has been done.

So this can be audio, video call or anything. But in between, you get the unencrypted call. This is not an end to encryption in this case. This is because you are doing it through the SIP server; this is technically feasible, which is why this is a legally accepted way of setting up voice over IP calls. So this is how you do the call tapping. We use a component called a media gateway.

(Refer Slide Time: 19:42)

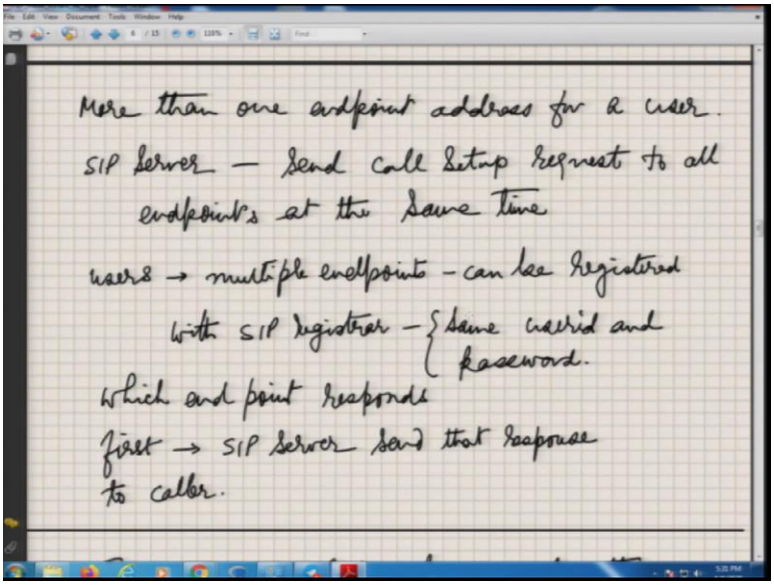


Media gateway also has more roles to play. For example, an IIT Kanpur, we have a media gateway. So if I want to call somebody outside IIT, I will talk to my SIP server. In turn, talks to the media gateway, which then talks to outside public switch telephone network, say BSNL, or we have Airtel as well as we have Reliance and Tata Docomo. So all three will be the PSTN network. So this media gateway will do all the signaling, all the call setup to destination.

And the reverse if everything happens, media call is being set up between media gateway and voice over and the telephone instrument. So this signaling is SIP. And between the SIP server and media gateway, there is a SIP can be used, or media gateway control protocol is also available at IETF, that also can be used. But from the media gateway to PSTN, it is mostly SS7 signaling. Even we are using even PRI systems here. They use SS7 signaling for communication.

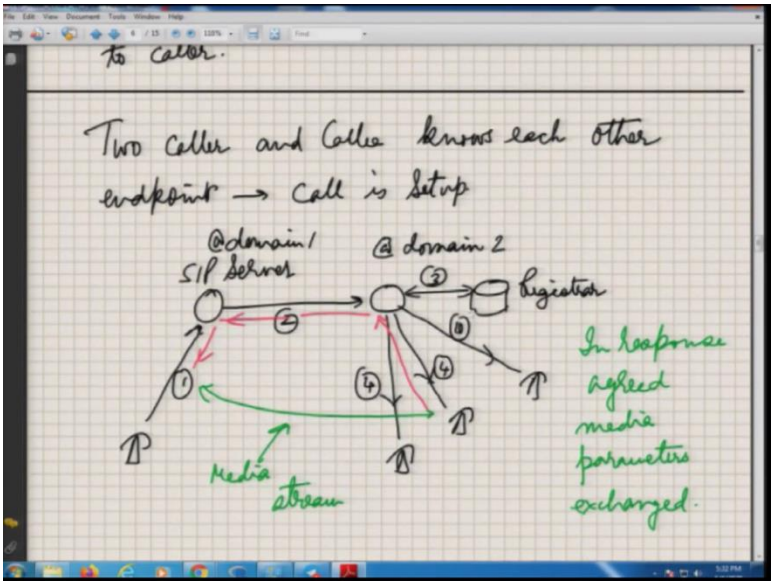
PSTN internally can act you to gatewaying or translation. Because most of the PSTN network in the, their core is using voice over IP now. This gives you far more efficient because of the statistical multiplexing; you are not doing circuit switching; you are now packet switching here, mostly in the back end.

(Refer Slide Time: 20:58)



And if there is more than one endpoint address for a user, the SIP server will send a call request to all the endpoints simultaneously. And all multiple endpoints need to be registered with the SIP registrar. The same user ID and password that is a key thing. And whichever endpoint responds first, the SIP server will send that response to the caller, and other endpoints are technically being removed from this particular call setup process.

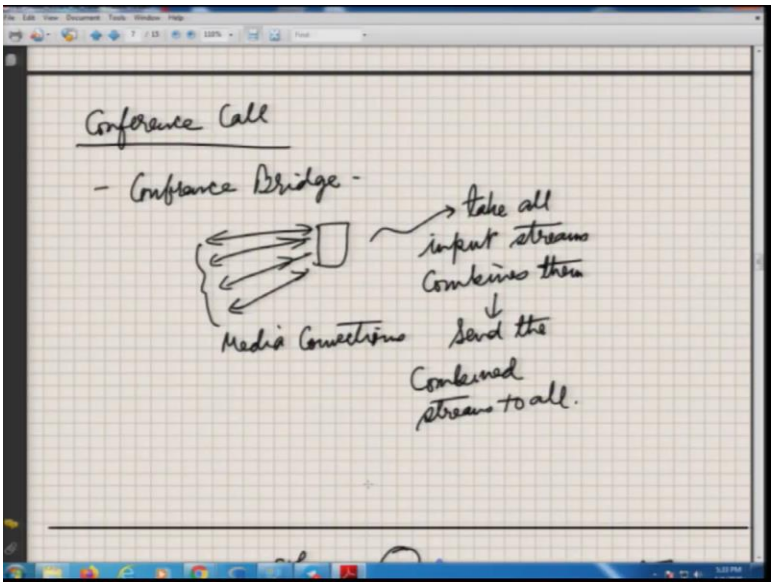
(Refer Slide Time: 21:27)



So this is the example of showing this, how these multiple calls will happen. You can see that with this black line, first, the call request goes. Call request goes to another domain, goes to the

registrar. Then we have these three devices which are being owned by the user. Step number 4 has been shown three times. The response goes there. Only one of the devices responds. And at this point, these other two devices are being removed from the call setup process. The response goes back, which is in the red line, to source the SIP server and go back to the client. And now the client knows to which guy we have to set up the voice over IP peer to peer call, and they set up a call, voice over IP call. So this all will be done with the mutually agreed media parameters.

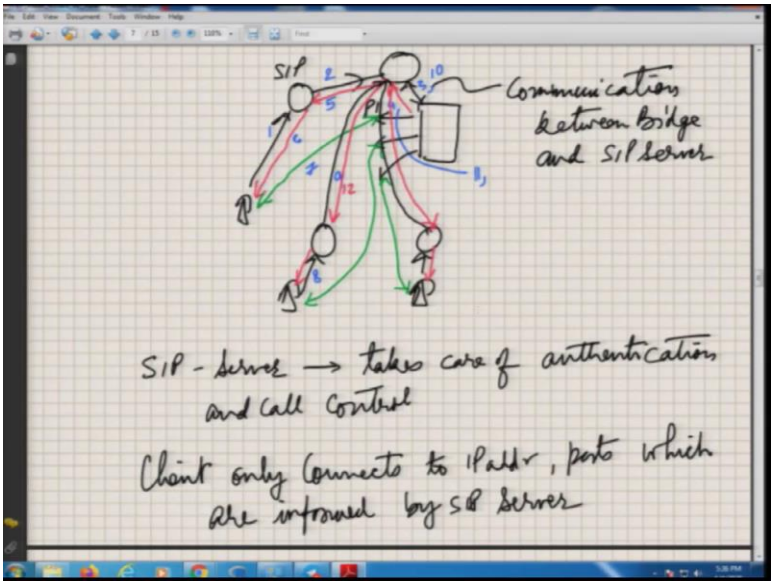
(Refer Slide Time: 22:21)



There is also called something called conference calls, which are done. When more than one person would like to talk, this is pretty popular now in the lockdown period, so people do kind of video conferencing and audio, so how that will be done. This is a tricky thing; mostly, it is done through a conference bridge. So you can appreciate this is a complicated thing.

So all participants have to send their audio and video to one single entity called Conference Bridge. This conference bridge will mix all these and create only one single audio-video string broadcasted to everybody. So everybody sees a mix of it. So anybody can speak, but everybody listens to whoever is speaking and whoever is all videos together. But more than a certain number of videos, of course, it becomes impractical on a small screen. So the user interface is important.

(Refer Slide Time: 23:13)



So this is how it is done. So here, for example, I am showing you there is a conference bridge. We set up communication through a SIP server, it goes to the SIP of the other domain where the bridge is there, and the bridge is being told that a call has to come in. You give me your one port IP address and port number; I should call it one port to connect user 1. And to user 1, the information is sent back. So user 1 connects to this particular port of the bridge.

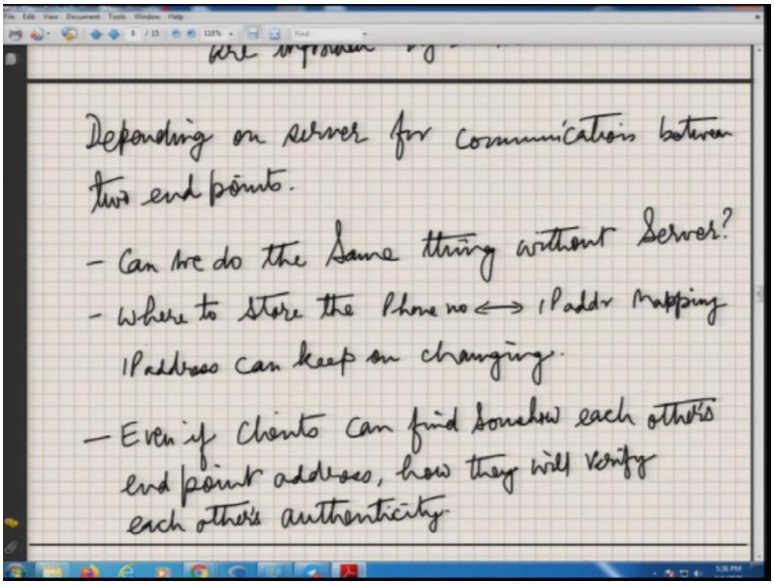
Now, when the call is being set up, we know that which are the other participants. So other participants are also sent the invitation. So these are their SIPs servers. So their SIP servers will now further send it to the users, users will confirm it, and this user, the confirmation when it returns to this SIP server, it will tell the bridge that this is a source, this IP address port number has to be used. This IP address port number has already been informed to this SIP server. So this particular call is going to be set up. This path is going to be set up. This bridge will do all mixing.

So this also a pretty standard feature. We also have in IIT Kanpur; we can have six phone calls, internal, external, can be bridged with our thing. So, doing a large number of these things, mixing becomes more complicated. But there are various ways and means by which this can be done.

So here, the SIP server mostly takes care of all authentication, all call control. It is not still a fully distributed system. They act as a centralized registry, for even setting up all this peer to peer

calls. There is even a configuration that audio-video of every peer will go to all other peers. You create a fully meshed connected network, and that is how you can also do a conference call.

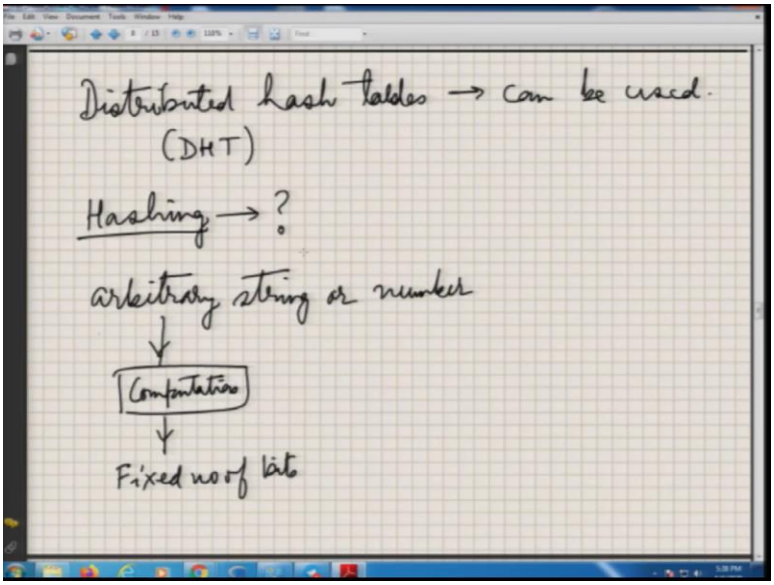
(Refer Slide Time: 25:11)



So depending on this, the server can have a server for communication between two endpoints. Mostly that is what we have been doing currently with the SIP server. But can we do this thing without a server? That is a question now. Can I do this? So we have to store the phone number and IP address mapping; that question remains. And this becomes extremely important if the IP address keeps on changing. So in our SIP server case, the IP address can keep on changing. Every time an address change is detected, it has to be reported back to the indexing server.

Even if you find out who the other guy is, how you will verify mutually that the other guy is a genuine person is not a fake guy? Because you are getting some random IP address and port number, you cannot be sure about that. So there has to be some kind of authentication. So the SIP server does to an extent by using domains and within the domain, using login password based registration mechanisms. So all calls are set up only through SIP servers; it is not done directly. We have to essentially now figure out a mechanism that can be worked without servers. That is a question which we have.

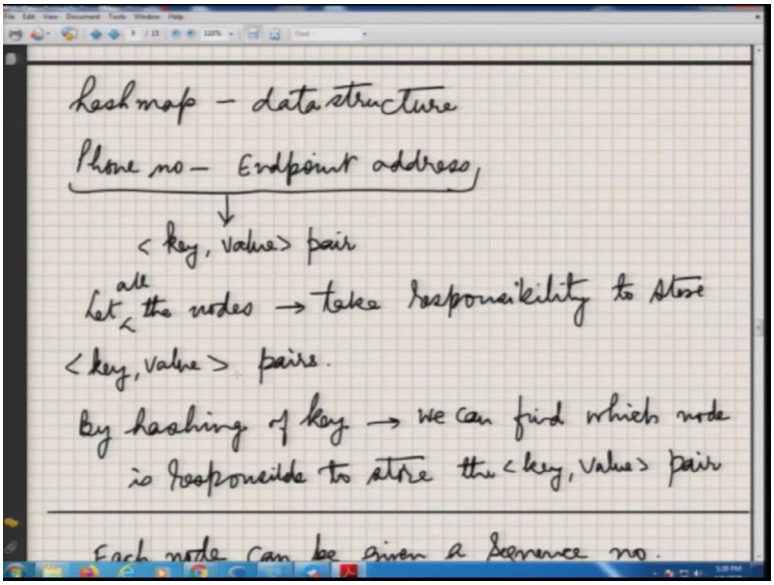
(Refer Slide Time: 26:29)



The question is if I am talking about without servers where I will store my IP address and port numbers to phone number mapping. The idea comes that we can use distributed hash tables; we call DHT. And there is a keyword here hashing. So what hashing does is that hashing is a specialized computation; it can be as simple as doing a modulo operation or a cryptographic hash. I will talk about it. So you will just put in any arbitrary string or number.

So normally, we always talk about the number here. It is not a string; it is a number every time for me. I just compute a 256-bit number or a 512-bit number or any large-sized number; I do the computation; I will get fix number of bits. So a cryptographic hash of 256 bit always gives me 256 bits at the output. Whatever be the size of the input, it does not matter. So its infinite size set to finite size, finite-size set mapping that is what it does.

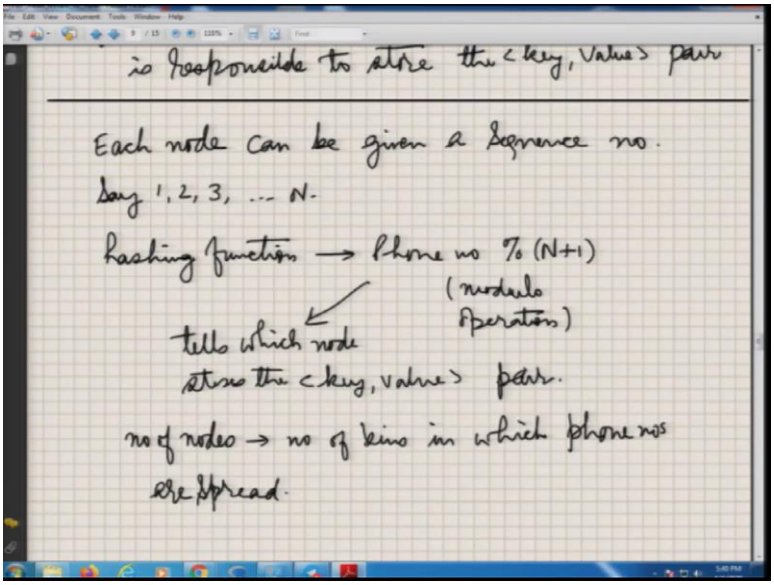
(Refer Slide Time: 27:32)



So I think, to better understand it, let us go to the hash map. Hash map is one of the popular data structures used. And the way it can be done is, I can have a phone number and endpoint address. So phone number is the key with which I have to search. The value for that phone number is the endpoint address, which is IP address, port number, transport, all those details, so that is a value. So I have to figure out the key, and from the key, I have to find out the endpoint value. So the key is stored where that is, what I need to figure out.

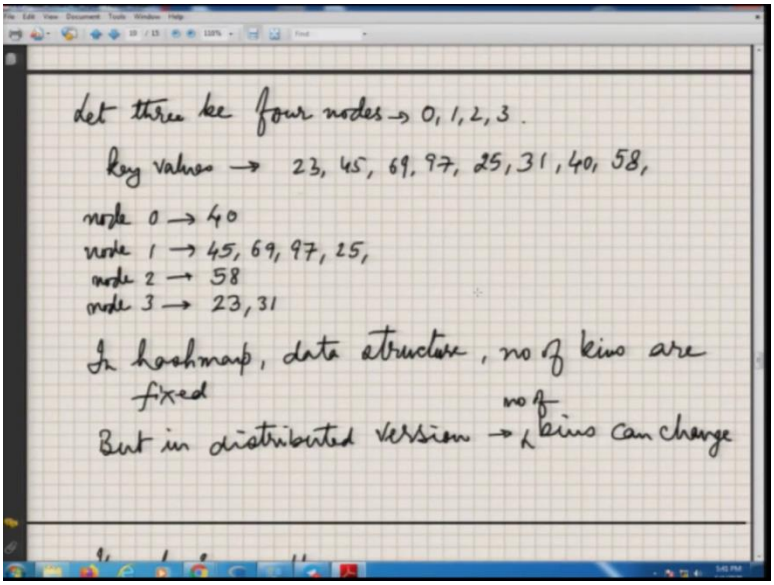
So if all nodes together, there is no central server; all nodes just take the responsibility to store the key-value pairs. And by hashing, and some number can identify each node. By hashing the key, we can find out who will be responsible for storing the key-value pair. So whatever after hashing, I will get the number, so that number, that particular node will be responsible for storing this particular key value.

(Refer Slide Time: 28:33)



So I am giving a very simple example that the hashing function can be a modulo operation. So you can give each node a number 1, 2, 3, N. So whatever all possible key values, I can do the phone number's modulo operation. With N+1, I will get the number ranging from 0 to N, so there should be a 0 here. So my apologies for that. There should be 0, which should be put here. From this, we will figure out which node is supposed to hold this particular key-value pair. And I can just go to that node, but remember, I am assuming I know how many nodes are there; I know each node ID. We will now be generalizing it, where you need not know all nodes in the network. You only need to know a part of them and still maintain the connectivity. These kinds of you are creating N bins in which the phone numbers have to be spread out.

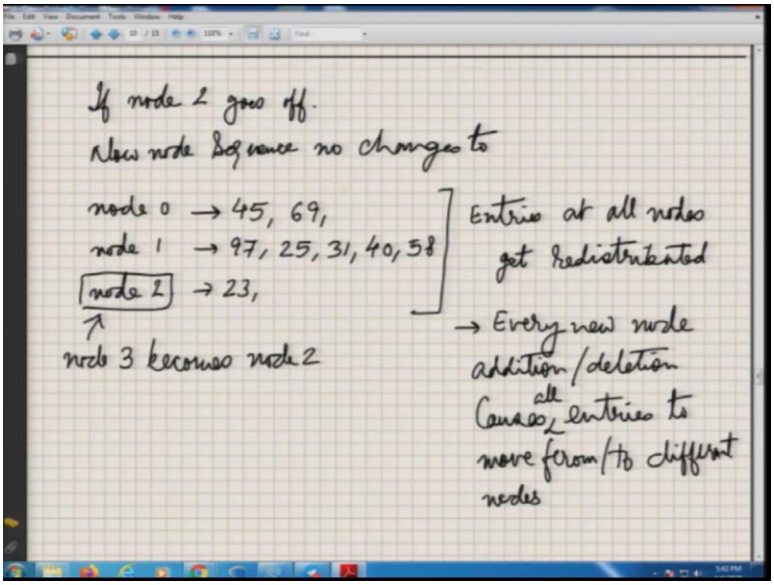
(Refer Slide Time: 29:23)



So let us take a case, there are four nodes only, 0, 1, 2, 3. And these are the key values. These are technically 4, 2 digit phone numbers. For 23, you will be responsible for holding up the value, basically the IP address, the port number, and transport. So if I get 23, I can do a modulo four operation here, and if I do modulo four operation, so 23 if I do modulo 4, it turns out to be 3. So node 3 is supposed to maintain 23; the value, the key-value pair for 23, is held here. For 31, it will be held here because 31 by 4 is the modulo. If you do, you will get 3. So 40 will be with node 0. 45, 69 will be here. Wonderful. So I can distributively figure out a unique location where the key value has to be stored.

So in the hash map data structure, which is used in computers in algorithms, this is used. So the number of bins is mostly fixed. But in a distributed version, nodes can go in and out; the node can turn off. You might end up in 0, 1, 3. Only three nodes, so you have to renumber them as 0, 1, 2. Then what is going to happen? Let us see what happens with that.

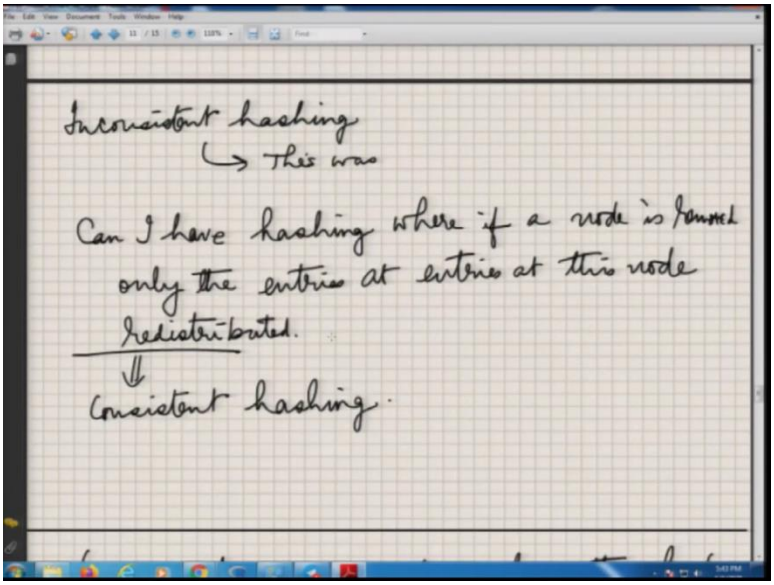
(Refer Slide Time: 30:44)



If the number of bins changes, if node 2 goes off, now the node sequence number will be changed to 0, 1, 2. So 0, 1, 2, 3 is, so 2 is gone, so 3 will become 2. So all the entries now have to be redistributed. So if I do it, 23 will move on to node 2, which was earlier at node 3, so node 3 has now become node 2, so it is not moving anywhere. 31 has now moved to node 1.

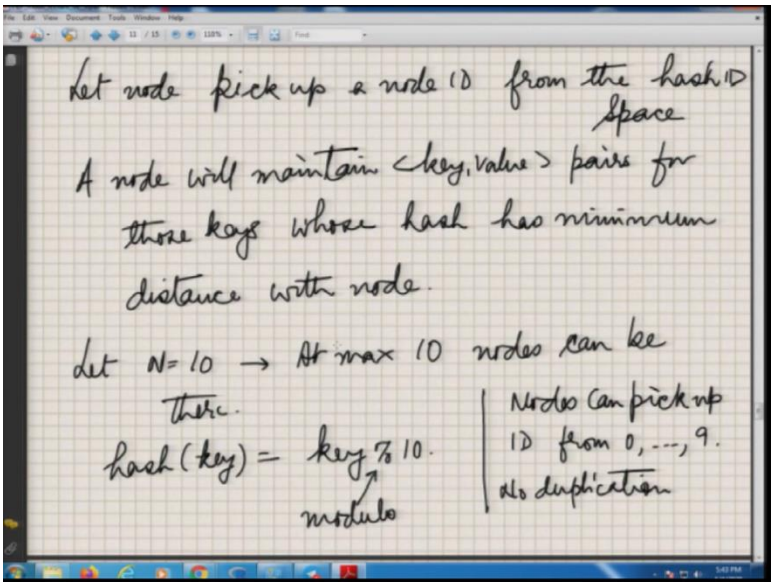
So you will figure out from every node almost every entry is moving here and there. There is a lot of movement around. So every new node addition or every deletion will cause redistribution for all entries. Now, this is not acceptable. This will create a kind of a storm every time. A node comes in, or a node goes out in a distributed system. This is what we call inconsistent hashing.

(Refer Slide Time: 31:37)



So can we have a hashing where even if a node is removed, only the entries at that nodes will get redistributed uniformly across everybody? So if you can do this, we get something called consistent hashing.

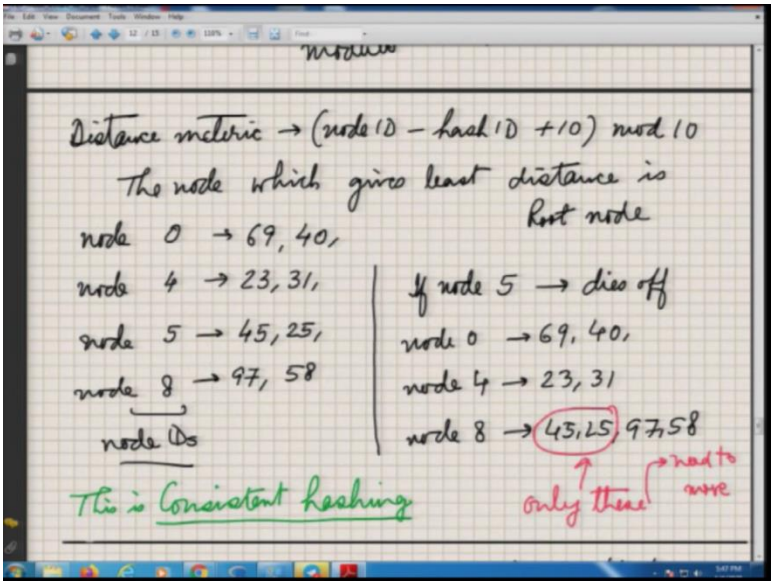
(Refer Slide Time: 31:54)



So, in this case, we will let each node pick up a node ID from the hash ID space. So our hash ID is specifically defined, and then each node will pick up some value. And a node will maintain key-value pairs. So it will take a key, the key will be hashed, and this hash has a minimum distance from this particular node responsible. So that is the idea of consistent hashing.

So if a node dies off, all the entries in its vicinity will tend to move to some other nodes now closer to them. So the nodes, the entries at other nodes, will not be moving anywhere because those nodes remain the closest to them. So that is the idea. Let us make N = 10, so the number of nodes and a maximum of ten nodes can be there. Hash of the key is still is the percentage of 10. Nodes can pick up the ID from 0 to 9 now. There is no duplication. This is a hash ID. And distance metric, the way I define is (node ID - hash ID +10).

(Refer Slide Time: 33:12)



So if the node ID is 9, the hash is said 10, so in that case, 9 will not be the closest, it will be 9 minus 10, it will be minus 1, plus 10, so it will be, and some other node which is going to be next in a cyclic fashion. So if 0 nodes exist, so 0 minus 9 plus 10, it will be a plus 1, which is the smallest value which will come out. 9 will not, 8 will not be the closest one, but the 9th one will be there. 9th will not be the closes, but 0, node 0 will be the closest one. So node which gives the least distance will become the route node. So that is the definition of route node which we use. So any node which is the closest node, by distance metric that is a route node.

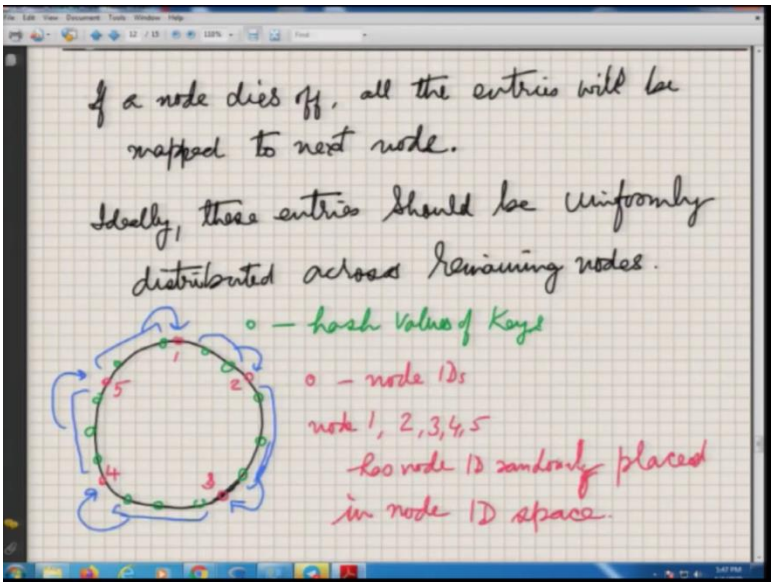
So for node 0, the route node will turn out to be 69 or 40. For node 4 it is 23, 31—node 5, 45, 25. So, you can verify this. So when you take 69, I have only node available is 0, 4, 5, 8. I just picked up some nodes, and I have given these IDs. So when I do 69 modulo 10, the hash will turn out to be 9 in this case. So 9 in a cyclic fashion. So 0 should be the 0, 4, 5, 8. Between 8 and

0, 9 comes out, so 0 should be the closest node to become the route node. So 4 is 40. 40's hash will turn out to be 0, which is this node ID itself, so this is the closest node.

For 23, 31, so basically whatever is the unit numbers here. This will be the root node for all 1, 2, 3, 4 in the unit place. 4, 5, this will be the root node for 6, 7, 8, this will be the root node. Then 9 and 0, this will be the root node. So they have got redistributed. If node 5 now dies off, what will happen? So this 69, 40 remains with 0; this remains with, and node 4 will remain.

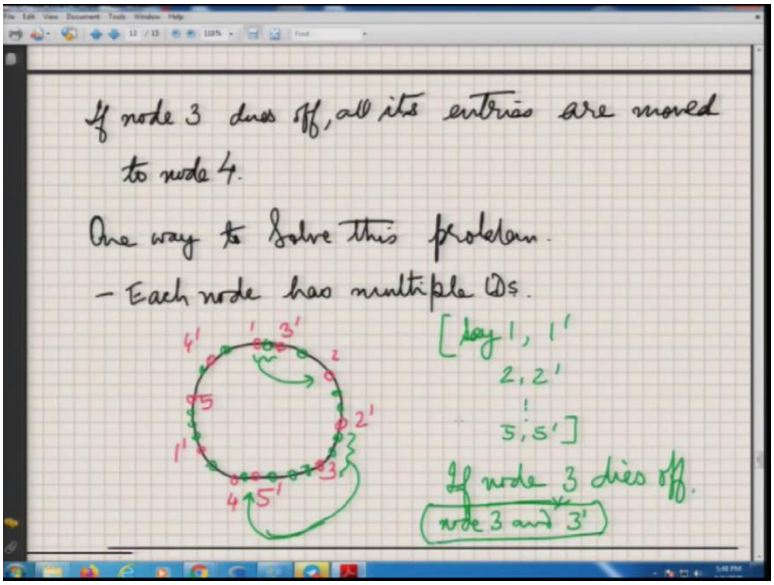
Now 45, 25 will be responsible, will now be the responsibility of node 8. Only these need to be relocated and the node which has failed. This is consistent hashing. But this has a problem; all these nodes have moved to only one node when node 5 has failed. They are not uniformly distributed. So it is a kind of a lopsided reorganization which happens.

(Refer Slide Time: 35:45)



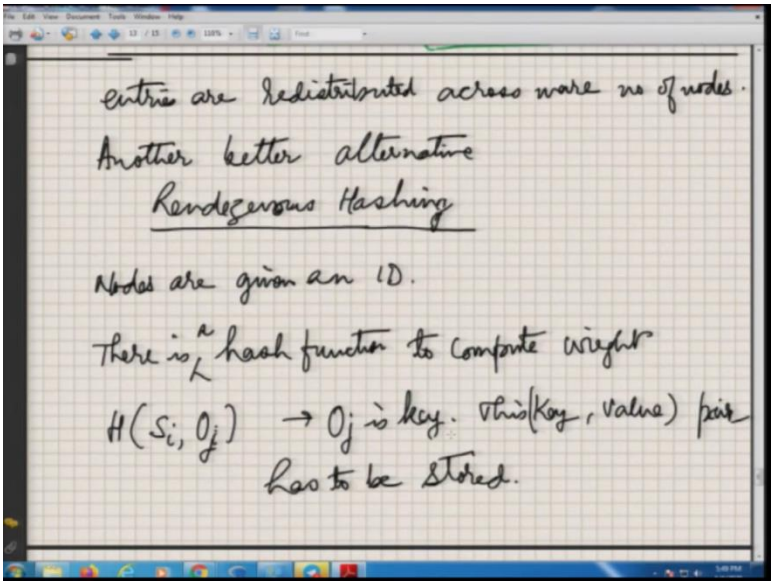
Now, if a node dies off, all the entries will be mapped to the next node. That is a problem here. Now, this is represented as the same thing in a cyclic fashion. This is the way we organize it in also chord algorithm. So I will talk about chord as we go along in the series of lectures. So if a node dies off, all the entries will be mapped to the next node. This is what is happening. I am giving an example if this particular node dies off, node 3, all these three entries will now become the responsibility of node 4. But this should be now, ideally should be uniformly distributed. How can this be done?

(Refer Slide Time: 36:26)



So this where I am showing that we can do something here. One way to solve this problem is that each node will have multiple IDs. So I can have 1-1 prime, 2-2 prime, 5-5 prime, and they can be placed with a different this. I am only giving the node number, these are not the node IDs, but they are randomly giving some node IDs. So each node is now being randomly placed at multiple locations. So, in this case, for example, node 3 and 3 prime, both of these will fail. So only these, these two nodes will be moving to node 5 prime, and this one node will be moving to 2 prime, that is it. So these are the more uniform organization, reorganization, which will be feasible. So that is one way of doing it.

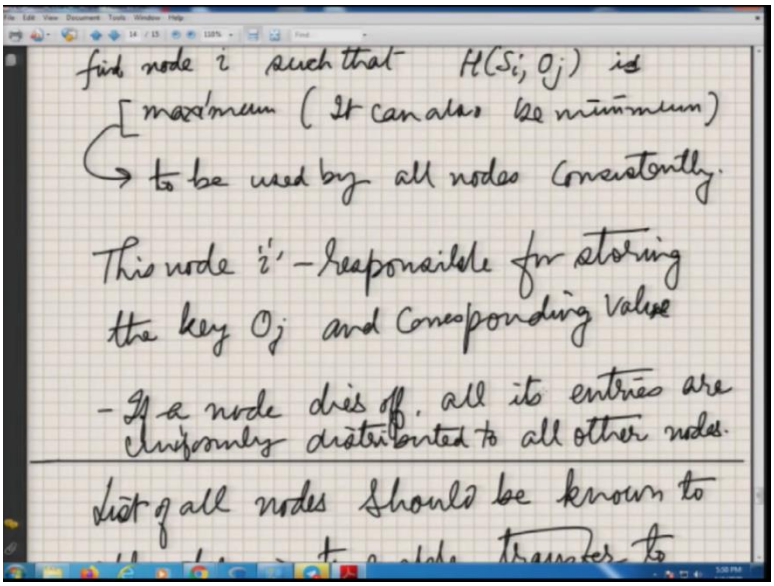
(Refer Slide Time: 37:15)

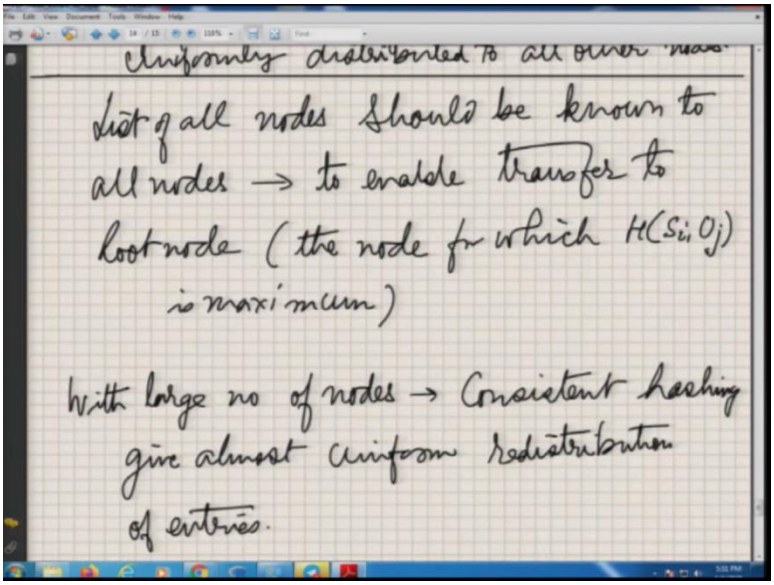


Then there is another way of something called rendezvous hashing. Rendezvous hashing came up when people were building a multicasting system. So when we want to create multicasting, we need to create something called a rendezvous node. Everybody who wants to join a multicast will always send a joint request towards the rendezvous node. So normally, if we have only one or two rendezvous nodes in the whole of the world, it will be a problem. So we need a collection of them. But which one has to be picked up by me that is a question. And if a rendezvous node dies off, only some of the entries have to be redistributed. But the idea is that every node knows how many rendezvous nodes are there and what are their IDs.

So normally all these, this we call rendezvous hashing. So nodes are all given in ID, and there is a hash function to compute this weight. So the hash function will have an input SI and OJ. OJ is the key in which you are trying to see which guy is responsible for it. SI is the node ID for the nodes which are participating, which are going to store. So I will pass these two node IDs and the hash of the key through the hashing function. I will get weight, and whichever guy has the highest weight or the lowest weight should be my root node. Either weight is okay. But everybody has to be consistently using this.

(Refer Slide Time: 38:35)



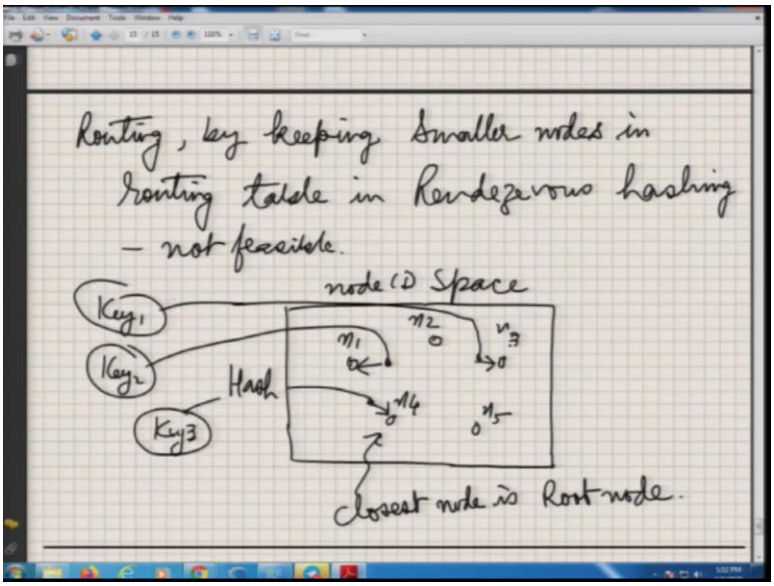


So node i is going to have the highest value, then it will be the responsible node. And if it dies off, all its entries will be uniformly distributed across all other nodes because of this randomize hash function. This randomized hash function essentially is known to everybody. They all compute in the same fashion; they all will come with the same result because of that.

This is essentially far better than the earlier one, but we cannot use it in DHTs. For reasons because you may not be keeping track of all node IDs, you cannot find out who will be a root node using this method. So we have to go with the distance metric based system. But we assume

that there will be an infinitely large number of people participating in the DHT. Even if one or two dies off, those entries will go to the next node or the closer node, maybe a few of them. So that differential which will be created, the much smaller. And these failures will be happening uniformly across the whole network. So it is a kind of a more or less uniform thing, so I need not worry about that uniform redistribution. With a large number of nodes, consistent hashing is good enough; we need not use rendezvous hashing in this case.

(Refer Slide Time: 40:00)



Now, this is the basic principle on which this will be working. So you will have key 1; it will map into the hash space to this location. Another key 2 maps to this location, key 3 maps to this location. Now, these circles are the node IDs. For each entry, each hash value of the key, I will find out which is the closest node and the responsible node. I call it the root node. So this will be the guy who will be holding on to the key-value pair for this particular key. So this is the basic principle that is used in DHT.

So in the next lecture, we will be exploring how this DHT will be used as a proper algorithm. First of all, we must understand how we ensure that the whole network always remains connected; it will never get isolated. It should never happen that a node with a certain node ID cannot be reached by any other certain part of the network. So you should be able to reach from any node to any other node in the network. Even if nodes fail, then also this connectivity is

always maintained. So that is what we require logarithmic partitioning for that. So most of the algorithms are based on that concept. So we will explore that in the coming lecture.