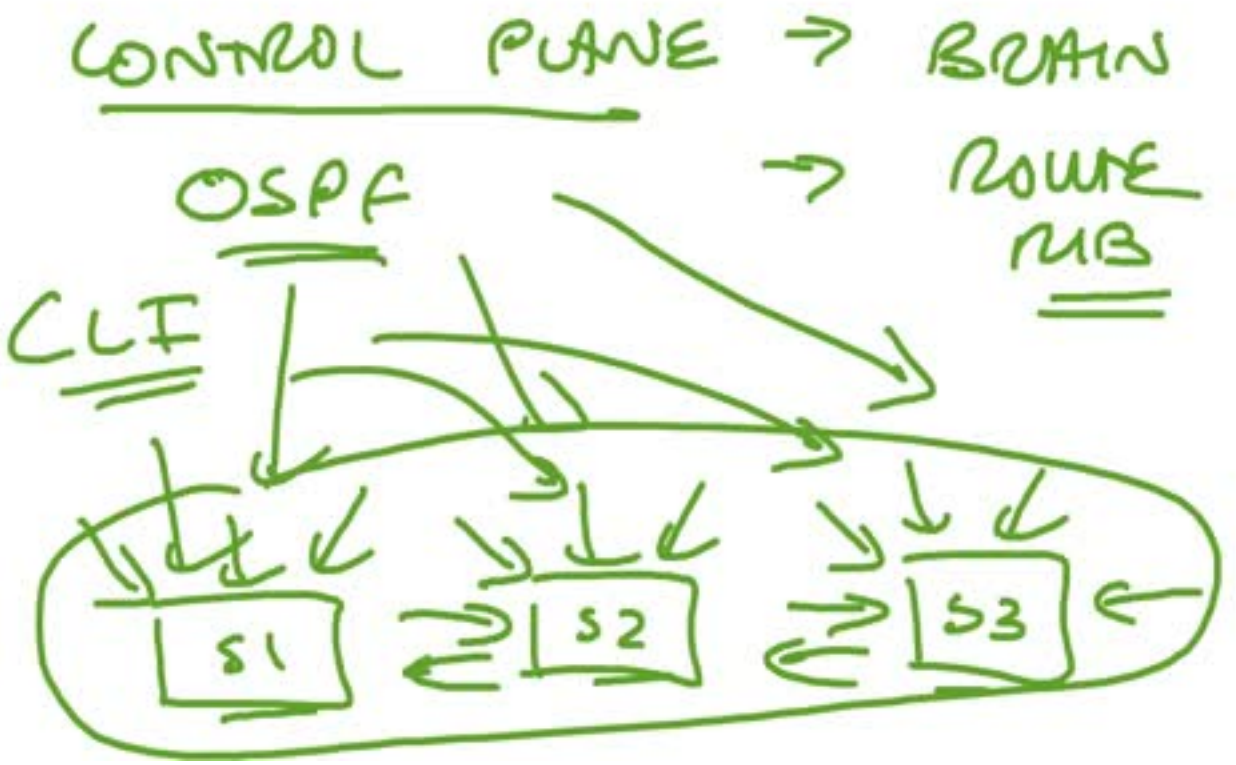
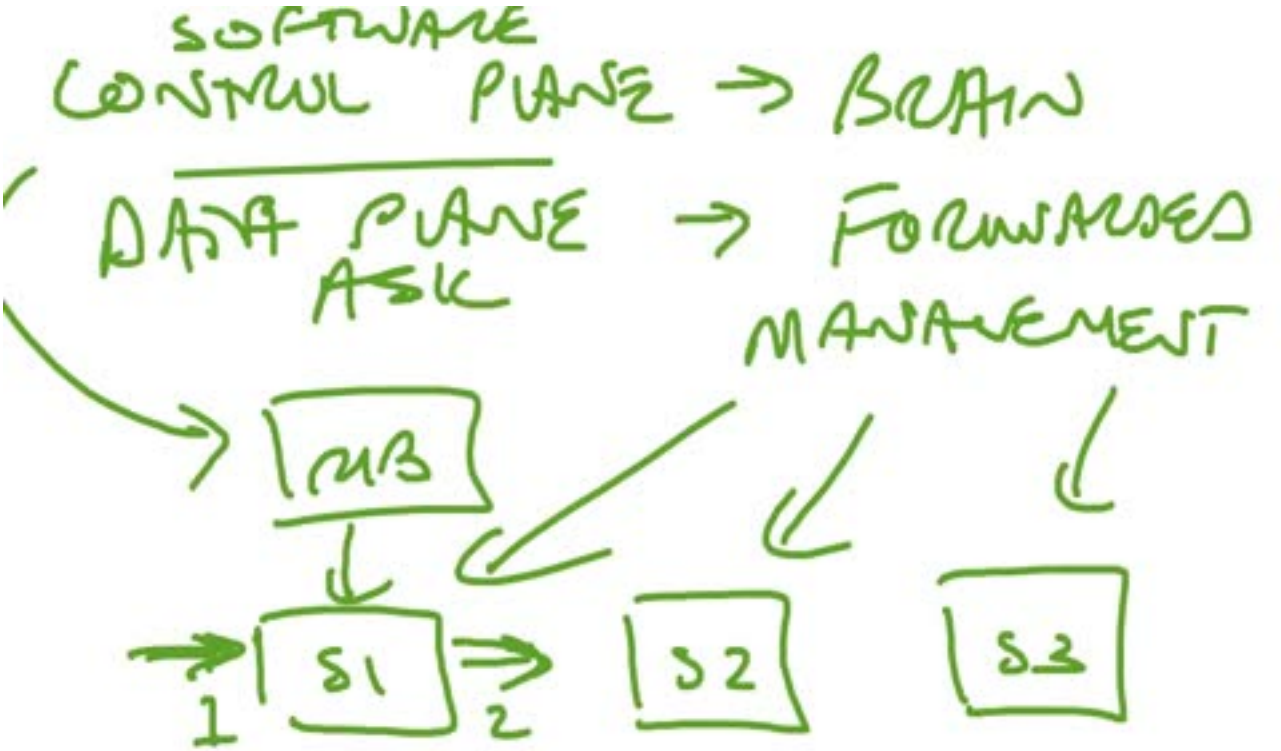
In this video we going to discuss software defined networking sdn and openflow now one of the hardest things to understand with s d n.
O software defined networking is the various definitions of sdn the gentleman who seen as the father of east end of software defined networking.
What sauced what is acn and he said well he's not sure anymore because it's definition has changed.
It's kind of like an umbrella term for various ways to use software to manage and manipulate networks for this discussion.
I'm going to explain three main.
Reviews of software defined networking the first definition is open.
Sdn typically using a protocol cold openflow the second definition is sdn via.
Apis application programming interfaces and the food way or so definition of sdn is sdn of vaya overlays.
The open networking foundation which is the organization in charge of the openflow protocol and openflow standard define tasty in as the separation of the control plane and data plane where the networking devices all control or updated using the open flood protocol.
That is one definition of sdm cisco's an example likes to use the sd in via api model wear functionality on networking devices is exposed using a rich application programming interface.
Api developers can manipulate network devices using a richer api then was possible using for instance the cli and snmp to netflix can be programmed using a richer or extended application programming interface vmware use sdny overlays wait for example vxlan tunnels are used across a network infrastructure for the atp course.
We going to concentrate on the open sdn model using openflow but i will explain the other visions of st and briefly in a traditional environment way for example a company has three switches three routers.
Each device has a local control plane and local data plan.
Now the control plane is essentially the brain of the device.
A device with a local control plane has for example a local a routing table or local rip routing information base particles like ospf all configured on these devices to update the routing table.
Open shortest path first ospf is a distributed routing protocol that is run on each device and exchanges routes between the devices to allow them to work out what the topology of the network looks like ospf is a complicated distributed routing protocol in a traditional environment the control plane runs locally on each device.
In other words.
Each device has a local brain advocates of sdn will say that this is a complex way to work out to the topology of the network.
There is no single device that has visibility of the entire network each device as to work out independently what the network looks like and together they synchronize a link state database which is the state of the topology at a given point in time.
If you have multiple network devices from a management point of view using the cli or command line interface you would have to connect to each device and individually and typically manually configure that networking device.

Each device also has a local data plan.
The date of plane is hot pockets of forwarded through the device.

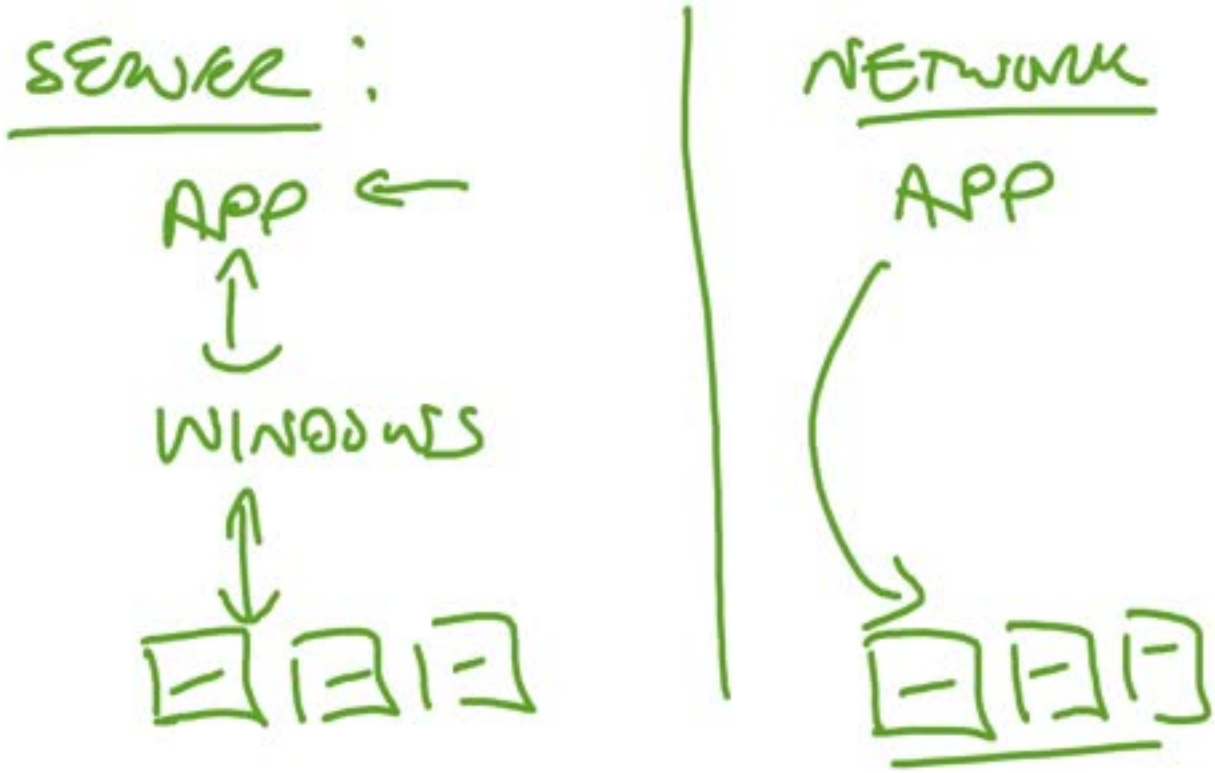


If you had three switches each switch has a local control plane the local control planes the local brain once again.
Each device also has a local the data plane.
This is used for forwarding of the traffic.
The how is the traffic forwarded through the device some other ways traffic around some pulled one.
It needs to be forwarded out of tea.
And that information is programmed using the control plane in a riding information bases an example or a mac address table where the device of loom bands with mac addresses on the topology that information can then be pushed into hard way into the data plan.
So the control plane is typically running software the date of plane is a copy of the routing table as an example running hard way to allow for high throughput folding of traffic from one point to another from a management point of view each device has its own management playing in other words you need to configure each device individually if you had a hundred switches or a hundred route is you would have a hundred control planes or hundreds shepherd brains.
You would have a hundred data planes and you have a hundred management in two faces to manage those devices you would need to town.
It will connect to the cli of each of those hundred devices to configure ospf how to configure spanning tree as an example.
Another problem that limits automation of networking devices is the fact that every device has its own proprietary operating system and own proprietary interfaces.
If this device was an hp conway switch and this device was an hp provision switch and this device was a cisco switch and you had some other devices from juniper and other networking vendors.



It's very difficult to create an application or the new rotting particle that is installed and every one of those devices example if i developed a new routing protocol david's dodgy routing protocol ddrp which i think is a very good routing protocol.
It would be very difficult for me to get a vendor's such as hp cisco juniper and others to run this new routing protocol that i've developed.
There is no opening to face on these devices that allow me to change the control plane and hence the data plan of the switches icon to develop a new routing protocol and just install it on these.
Switches the operating system on these switches is proprietary.
I have no access to the operating system icon change the operating system to install a new routing protocol icon change the api show interfaces on these devices because its proprietary so advocates of a sea and an open flow will say that this limits innovation it would make much more sense to open up the networking devices to allow other people to develop applications protocols and utilities that allow for rapid development and innovation in networking.
On the other hand when you look at server technologies and server development.
There has been rapid innovation in the last 10 years networking has been stagnant.
We still configure ospf in the same way today as we did 10 years ago but if you think about pcs will service i as an application developer could write an application any application something like microsoft word or a game was some kind of application that specific to a business requirements.
I would write that application for an operating system such as windows or linux will some type of os operating system abstracts the physical hard way from me as an application developer if i was running an application and installing it on an hp laptop or a dell laptop virtual machine running in vmware or in virtualbox many other types of physical implementations is an application developer do not need to be aware of the network interface cards used on these physical machine virtual machines.
I am literally write an application to an api for the operating system and the operating system hides

the complexities of the network interface card the memory.
And other physical characteristics of the hard way why is the application developer unaware of the physical server that my application is installed on the rapid development of vmware and other virtualization technologies have allowed the industry to extend this where the operating system doesn't even know which hardware it's physically installed on you can rapidly move a virtual machine from one hardware platform to another so in other words i could create a virtual machine on my laptop which may be an hp laptop.
Could you could install that on an apple mac running within avian way virtualized environment that is because of abstraction of the hard way from the application that doesn't exist in networking today.
However so when i survey environment i could create an application to an abstraction layer.
Operating system such as windows windows hides the complexity of the hard way that the application is installed on from me as the application developer.
I don't have to be concerned about the physical hard way that my application is installed on i can write my application in a high level programming language such as pie iphone or visual basic i am unaware of and don't need to be concerned about the physical hardware that the application is run on that is not possible today in networking.
I cannot create an application that's installed on networking devices because this is proprietary.
I have no access to the vendor operating system code to make changes.
So with open sdn and openflow the idea is to create an opening to face.



On networking devices and create an abstraction layer to allow for rapid application development.
So the situation changes to the following.
So what we introduce is a controller.
Abstraction layer the controller talks to the networking devices using an open standard protocol in this case open flow open flow provides an open api open interface to networking devices the idea.

Where is it doesn't matter which operating system or vendor the networking devices using there is an opening to face to managing the flow table of that device.
This is known as the southbound ati on the northbound api we have two interfaces one using the restful api and the other one using the java api these are two interfaces that allows an application developer to manipulate flow tables and flu entries on networking devices without communicating directly to those networking devices.
The application developer is extracted from the hard way and doesn't need to be aware of the details and requirements of physical asics on switches and networking devices.
The application developer just like in windows uses a high level api in this case of rest or java api to communicate to the controller and the controller takes care of the nitty gritty or complicated details of updating flow tables on switches.
So previously we had three switches each with a local control plane and local data plan in an open flow sdn environment.
We have my controller and in its purest form.

The brain of the networking device is moved to the controller.
So rather than the networking devices each having their own brain.
There is now a central brain in control of the networking devices.
This provides a centralized device for centralized control centralized policies and easy and management as an analogy and it's only an analogy in a wireless environment previously.
We had a ton of us access points.
He might have had a hundred of tournaments access points these days we have lightweight or controlled access points where a hundred access points are controlled by central crayola rather than each access point having its own local brain.
It's much easier to manage a hundred access points using a centralized controller.
Then it is to manually create policies on a hundred individual access points that's similar to the

idea.
Here we we have a centralized controller that is pushing policies or configuration changes down to networking devices.
It allows an application program at to create an application that can manipulate the flow of traffic on the network.
So i could create an application that sends a command through the rest api to the controller.
Which then updates the switches using openflow that now says that traffic arriving with a mac address of a going to mac address of be should flow through the network along this path.
Now they have some optional videos that i've added to this training.
We're actually show you a application in this case is very simple application of bash script updating the flow tables of the switches and manipulating the flows of traffic in the network that can be extended to fall more complicated applications and i'll talk about some examples in the moment but to summarize in open sdn sdn using openflow the networking devices no longer have a local brain.
So in its purest form superior open sdn the brain is removed from the networking devices and he's put into a centralized controller the switches use what is cold a flow table that contains flow increase flow in trees are dated by the controller at the controller is an actual fact just a kanji which one abstraction layer that allows application developers to manipulate the flows in the network.
The power of sdn openflow comes into play with applications.
You are i could write an application that sends commands to the controller which in turn uses openflow to update the flow tables off the switches which in turn manipulates the flow of traffic in the network environment that's very difficult to do in a traditional networking.
Environment i just to reiterate terminology and then i'm going to talk about an example application at hp of developed.
We have an application.
It's uses the northbound api to communicate to an update information on the controller.
The controller uses the southbound api to update of the flow tables of networking devices.
Typically we talk about switches.
But these in the greater vision of things could be routers load balances by wolves and other devices northbound api typically uses a restful api or a java api to allow changes to the networking environment on the southbound api in a pure open flow environment.