```
var theJSON = edgeJSON;

System.log("JSON secondary output :" + JSON.stringify(theJSON, null, 2));


ipAddress = theJSON.vnics.vnics[0].addressGroups.addressGroups[0].primaryAddress;
GatewayName = theJSON.name;
var len = GatewayName.length;
//I just want to shave off the ESG- string part
var justName = GatewayName.substring(4, len);

System.log (ipAddress);
System.log (justName);


//The type of HTTP call
var theType = "GET";

operationUrl = "/2.0/vdn/virtualwires/"

//Set up and make request/PUT
var request = restHost.createRequest(theType, operationUrl, "");
request.setHeader("Content-Type", "application/json; charset=UTF-8");
request.setHeader("Accept", "application/json");

//Log the URL
//System.log("request: " + request.fullUrl);
var response = request.execute();

//log the JSON response
System.log("vdn response: " + response.contentAsString);
```

```
    // The value of 'response.contentAsString' is a 'String' type - you need to convert it to a JS 'Object' type
using JSON.parse

    var wiresObject = JSON.parse(response.contentAsString);

    // Now you can access the content of the Object referencing the members the way you want to
    var items = wiresObject.dataPage.data;

    // You can also convert things back to a JSON String using this

    System.log("The output of all Virtual Wires is captured here :" + JSON.stringify(items, null, 2));


    // walk the items list and find the entry with the name we want, then return the entry

    function findVWireByName(vwireName, vwireList)
    {
    // iterate over the list and find the one with the name we need
    for each(var vw in vwireList)
    {
        if(vw.name.indexOf(vwireName + "-vWire") > -1)
        {
            return vw;
            System.log("Virtual wire from Listing is : " + vw);
        }
    }
    return null;
    }

    // use the name building code to compute the wire name from the found item

    function buildWireName(vwireObj)
```

```
{
System.log("Computing vWire name from the JSON Object - " + JSON.stringify(vwireObj, null, 2));

var wireName;
var part1 = "vxw";

// check the backing data is present - the undefined you got previously might have indicated that
// some the data was missing for some entries?
if(vwireObj.backing != null && vwireObj.backing.length > 0)
{
var part2 = vwireObj.backing[0].switchObj.objectId;
}
else
{
    // CHOICE TO BE MADE HERE! Pick one strategy to suit what you want

    // normally, I would throw the error to kill things fast
    // throw "Failed to build the vWire name - backing was not specified!!!";

    // The other option is return null from this function and let the caller line deal with it...
    return null;
    // Use System.log(JSON.stringify(vwireObj, null, 2)) to print it to the logs so it can be inspected
}

var part3 = vwireObj.objectId;
var part4 = "sid";
var part5 = vwireObj.vdnId;
var part6 = vwireObj.name;

System.log(part1);
System.log(part2);
System.log(part3);
System.log(part5);
System.log(part6);

wireName = part1 + "-" + part2 + "-" + part3 + "-" + part4 + "-" + part5 + "-" + part6;

System.log("Computed vWire name is " + wireName);

return wireName;

}

//Identify and Build THE WIRENAME:

// Find the wire data we need first...
var myVWireItem = findVWireByName(justName, items);

// Check if we got it..
if(myVWireItem != null)
{
    // generate the name using the data we found
    var myVWireName = buildWireName(myVWireItem);
    if(myVWireName != null)
    {
    System.log("vWire name is " + myVWireName);
    }
    else
    {
    System.log("Failed to compute vWire name - some data was missing?" + JSON.stringify(myVWireItem, null, 2));
}
}
else
{
    System.log("No virtual wire found with name starting " + justName);
}



//Return the following Object back to vRA

var object = DynamicTypesManager.makeObject("TIE", "TIE Gateway",justName ,justName, new Array());

object.setProperty("objectId", id);
object.setProperty("GatewayName", GatewayName);
object.setProperty("ipAddress", ipAddress);
object.setProperty("NetworkName", myVWireName);

resultObj = object;
```