Threat Modeling - Scaling Recommendations and System Mapping

By: Chaocipher

Date: 10/28/2018

Target audience:

- ✕ Senior IT Management
- ✓ IT Management
- ✓ Technical Staff
- ✓ IT Vendors

Executive Summary

In this article we'll be looking at some recommendations for scaling your threat modeling program and a deep dive into how to effectively diagram a "system" and make it understandable, and convey a lot of information efficiently.

Table of Contents

Introduction

In this document we'll be focused on proper threat modeling that conveys a lot of information very quickly that can be used for both in-house developed systems or Commercial Off The Shelf products. In threat modeling, security teams must strike a balance between effective risk mitigation and project velocity. All project teams want to deliver their product fast and without security teams getting in the way. Security teams need to identify risks before the company is encumbered with them. Catching problems early in the planning stage can save an organization a lot of time and effort which usually means a lot of money as well.

Challenges

- Consistency.
- Lowering the impact to the project timeline.
- Adjustable to any type of system.
- COTS/SaaS vs. in-house.
- Labor cost of maintaining a threat modeling program.
- Output that has value beyond the current moment.

Types of Threat Modeling Programs

There are basically two types of threat modeling programs:

1. Compliance based
2. Comprehensive based

Compliance based threat modeling program can be built and constructed to support a compliance only risk assessment process. This program is designed to perform the minimum checks by using a base set of security controls and evaluating if those controls are in use. There is some value for auditing and in creating a sustainable process.

Second, a comprehensive based program can be built and constructed to perform, complete and valuable, risk assessments. The value of a comprehensive risk assessment process is that it will create a better security posture and will help fill the gaps in the effective enforcement of security controls. It also helps to better identify your layers of security.

If you want to run the first type of program, the compliance only risk assessments, then there are plenty of free tools to get you through that process and we can part ways here. This paper is targeting security programs that want to make an impact on the security posture of systems within their environments.

Challenges

Challenge: Lowering the impact to the project timeline.

To be effective at getting security relevant answers out of designs and project teams you must follow a regimented process for each engagement. How many times have you been asked for the "the security questions"? The project team just wants a copy of them so they can lie through each answer as quickly as possible and move on. Oddly, enough threat modeling is not where you catch teams in lies, so don't worry about that. Those lies are found after build, during the accreditation process. The goal of the security review here, is to perform a sanity check on the whole design.

Challenge: Adjustable to any type of system.

There are some tools out there that are very narrow in scope and do a great job but the real challenge is to design a template and technique that can be used on almost any system. One that doesn't exist or is variety of IaaS, SaaS, PaaS, and some other aaS that I haven't thought of. Over time you can create slimmed down question sets for recurring, well-known, designs.

When diagramming I recommend almost a blank sheet to start from. Just the legend and title areas should be there. I've found over the years that trying to preload the diagram with objects just leads to wasted effort removing or adjusting the size of objects on the diagram. The worst is when you forget to remove something that is not valid for the design. My recommendation is to build your custom stencil and load it up in Visio when you begin. This will allow for quick access to drop your shapes onto the diagram. This allows you to be very flexible in what you can draw and you don't have the added labor of maintaining too many templates that end up being only for snowflake designs.

Challenge: COTS/SaaS vs. in-house development.

What I see very commonly is a data flow diagram in a threat model. If you look online you find many examples of data flows inside of a computer. That's because the majority of the market has been putting out this type of content. That's fine for in-house developed applications. However, if you deal with COTS (commercial off the shelf) software, you are not authorized to see or know how that data is moved within the system. The same goes for SaaS models. You'll be given some information, but you'll likely not be authorized to know the behind the scenes stuff. Remember, once a computer is owned it's game over already. So, it rarely matters how you move the data around inside of a computer.

Challenge: Labor cost of maintaining a threat modeling program.

The cost of maintaining a system like this is quite large, but the payoff is huge. So, let’s go through some pros and cons:

Pros:

- Less security flaws being introduced into the environment.
- Better ability for teams to meet timelines through the central point of contact for project managers. Everyone that has a process that’s needed for new systems gets a workflow spun off.
- All teams that perform build work should now have a common library to confirm build work.
- All incident response teams should reference the same library to know how something was built for troubleshooting.
- Pushing security review earlier in the process of design enables teams to keep effort from being put into designs that would not be approved later on.
- Systems will have a lifecycle that can be leveraged to forecast upgrades and technical debt.
- The ability to rate aggregate technical debt and risk.
- Ability to quantify much of the IT inventory.
- Ability to respond to auditor inquiries regarding many aspects of the environment.

Cons:

- This work requires fairly technical and expensive staff to perform the assessment and documentation work.
- This work will slow down the building of systems at first. Over time it will be better for IT as the process matures and is integrated into the culture. If you take into account, how many IT projects fail or lose tremendous amounts of labor hours because of design flaws, it becomes clear this isn’t a slow down. However, from the point of view of each project team it can feel that way.
- To get the full benefit out of the process you have to apply the correct tools. Those tools can be time consuming to set up.

Challenge: Output that has value beyond the current moment.

It’s important to use a database to document these systems. Once you’ve committed to that element you can begin to mine the data for things like, “How many critical applications are using ServerX?” These answers can be very beneficial to security and operations systems. Another question such as, “How many applications require Chrome Browser on the clients?” Knowing this information can help IT leadership understand how much of the fleet needs to be upgraded and how to prioritize the efforts. Conversely, you could reverse the question and ask, “Which applications are keeping us from removing Adobe Flash from our environment?”

These answers are massively valuable to the organization. This information is usually siloed or not available at all to other teams that could use it.

Challenge: Diagram Consistency.

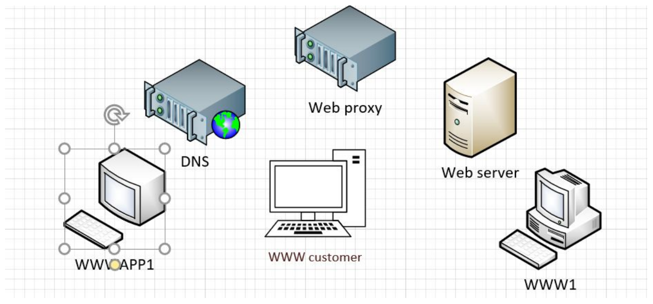
Challenge Description:

A very important part of diagramming is that we use consistent stencils, techniques, and level of detail. Also, it’s important the authors are using the same methodology for diagramming. The subtlety here is that those authors should be security people, not project managers, or enterprise architects. They aren’t security trained and will draw the diagram with the wrong focus.

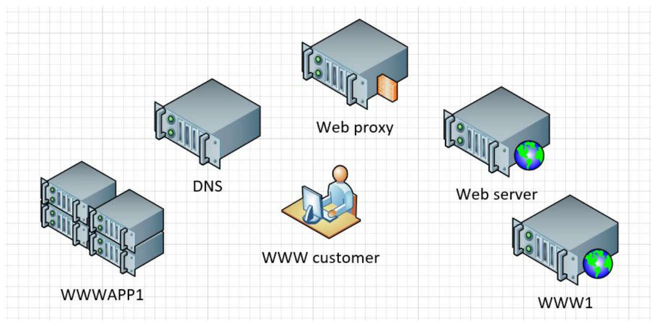
Challenge Diagram Example - All nodes using the same design palette:

Use consistent design palettes and sizes. It makes it easier to read and understand and looks more professional. As a security person I want to be able to determine clients from servers very easily. It's nice to make labels but that forces the user to read all the map at once rather than taking it in one layer at a time.

Bad way:



Better way:

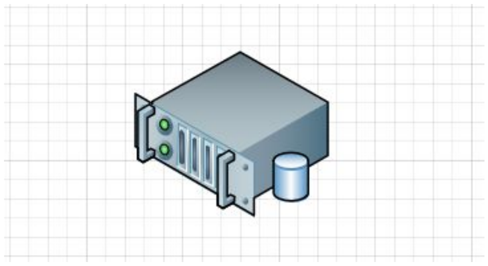


Challenge Diagram Example - Don’t use photos in diagrams:

Sometimes a stencil is not available and people will turn to pictures of the objects. This is bad for a couple reasons, firstly, it inflates the size of the file and that can cause other problems downstream. Pictures sometimes lack contrast when printing them at a small size. They also use more ink generally than an icon. Pictures lack the functionality needed for flexible connector terminating in older versions of Visio. This also breaks the benefits of consistency that we described earlier.

Bad way:

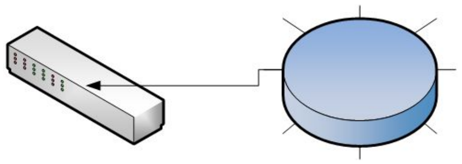
Better way:



Challenge Diagram Example - Terminate all lines and make sure they're visible:

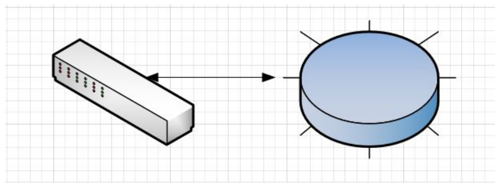
When you've completed your diagram make sure all the lines you've drawn terminate to an object and are connected. You'd be surprised how many times I see lines that don't go anywhere. Also, it's important to terminate the line so that if the object gets moved later on, maybe to make room for a new object, then the line will stay connected. Also, it's vital to make sure that the end of the line is visible to the reader. Some stencils are not build correctly and are really a collection of grouped objects and those are notorious for not terminating correctly in a spot that's visible; I'm looking at you AWS. Update: the new AWS Icons that were just released October 2018 are better in this regard. Arrowheads should flow of information too. This is backed up in the security plan document, but it starts here with quick information dissemination.

Bad way:



This line has two arrow heads!!!

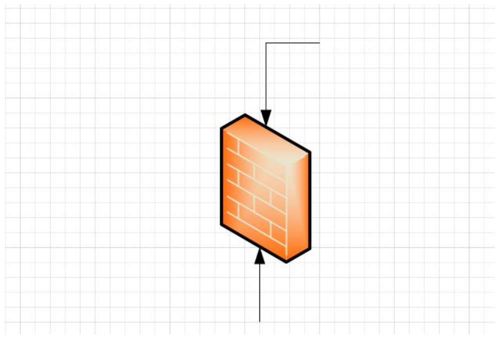
Better way:



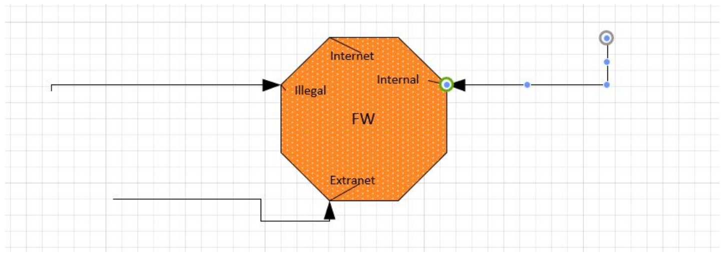
Challenge Diagram Example - Firewall Connections:

This issue is with the default firewall icon in Visio you can diagram the connections in a way that it becomes unclear where the traffic is going. The connected node needs to be clearly designated what network it's on and that the traffic is passing through the firewall.

Bad way:



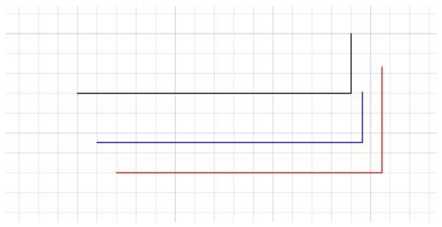
Better way:



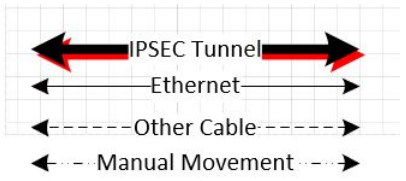
Challenge Diagram Example - Line colors and Dashes:

Many people use coloring to provide context to a link. This is bad for a couple reasons. Firstly, people who are color blind cannot see those colors clear enough or not at all. Using a couple forms of dashes is fine but those have a weakness that a line too short doesn't clearly show the pattern so don't do too many. Shadows can be used to help, but recently there was a bug in Visio 2016 where the shadow was placed too far away, so be flexible with that. Also, build a legend that will be consistent on all diagrams.

Bad way:



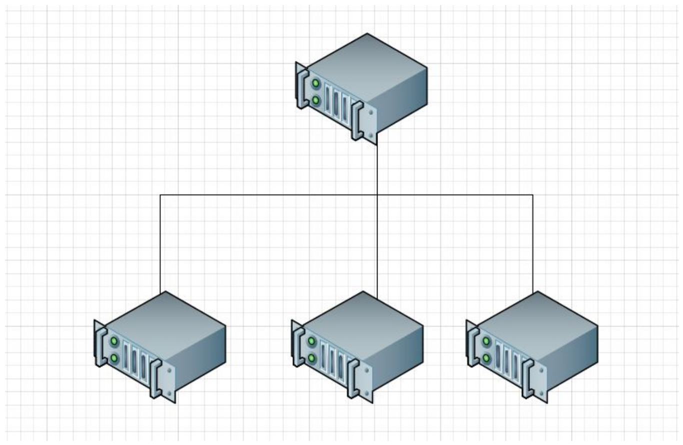
Better way:



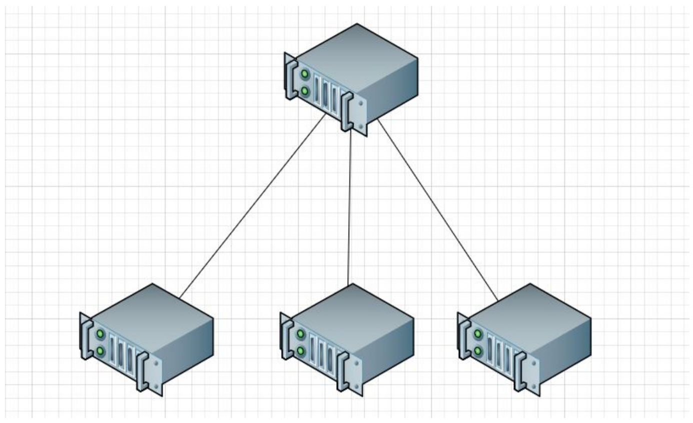
Challenge Diagram Example - Line shaping:

For security the most important piece of information that we need to convey in a diagram is how the data is getting from point A to B and what point A and B are. I've seen so many diagrams that share line paths. This kills the idea that I can know what information is going where. If the lines are redundant get rid of the extra lines.

Bad way:



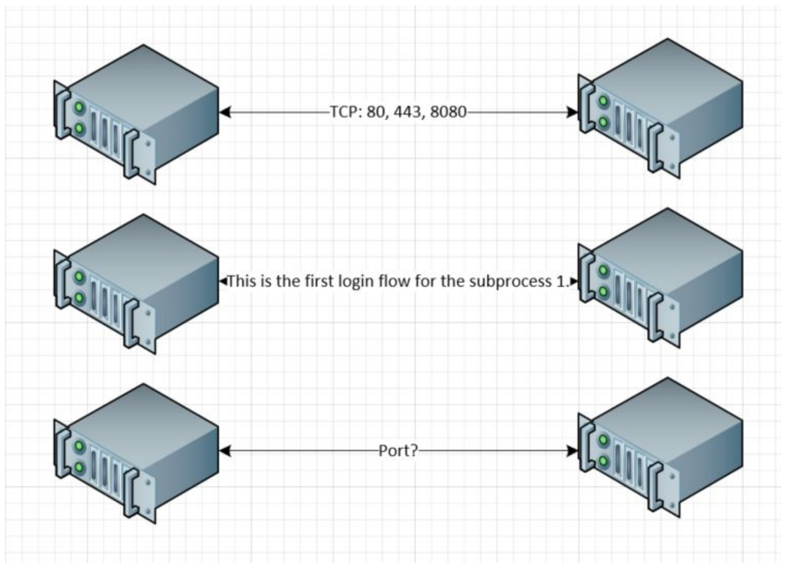
Better way:



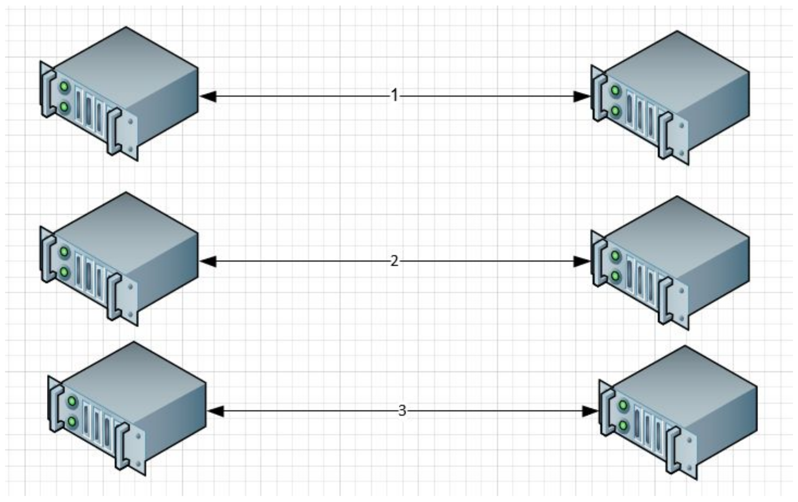
Challenge Diagram Example - Keep the flow labels simple:

People have many different ideas for what go on a link label. I'm to say don't do it. Label your link with a number that corresponds to a table that describes in detail what's going on. You have to do it anyways, so just keep it simple here.

Bad way:



Better way:

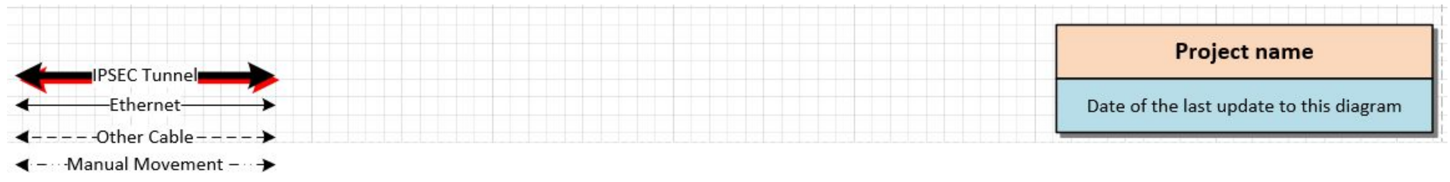


Challenge Diagram Example - Add a Title, Date, and Legend:

It’s a good habit to put a title, last modified date, and a legend on all your diagrams. Remember that in print form the name of the file doesn’t help you. Also, as a side note there are entities out there that require this information for submissions to their system. For example system designs that are submitted for criminal justice information system(CJIS) access require this information on all diagrams.

Bad way:
Nothing.

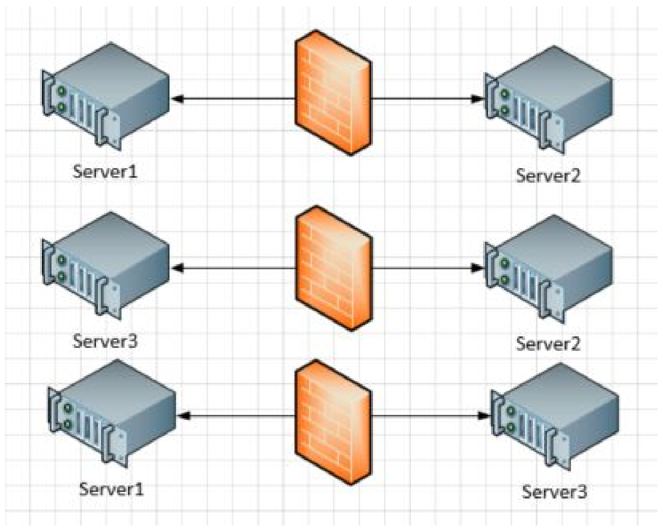
Better way:



<u>Challenge Diagram Example - Use unique names for all entities:</u>

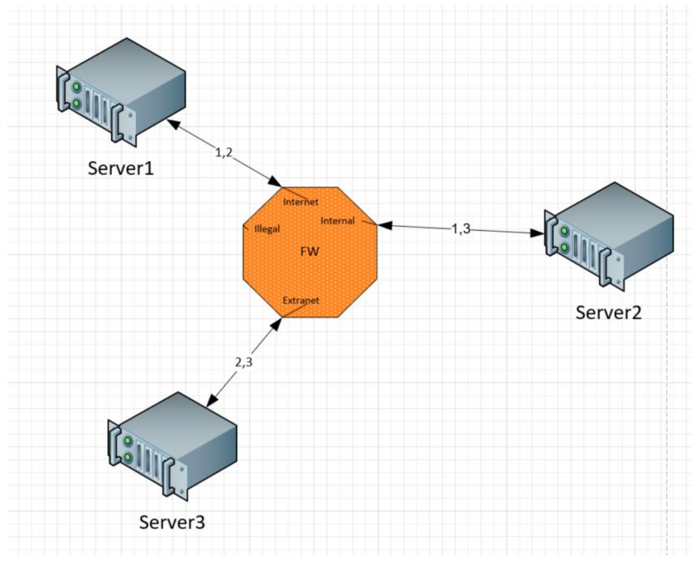
This one gets challenged from time to time, but believe me it will help reduce confusion. When collaborating on a diagram it's easier to call out items for discussion if they have unique names. Also, it dispels the idea that it could be the same object. This example below is a real world example that I had.

Bad way:



Several servers were on the map multiple times and the firewall was the same firewall. However, the diagram that I was given made it look like there were two times the servers and three times the firewalls.

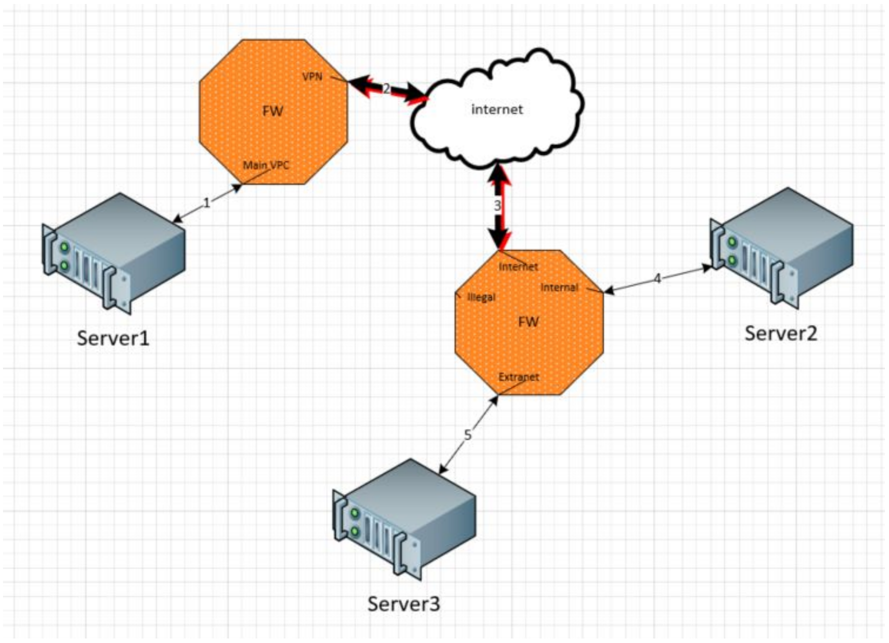
Better way:



Challenge Diagram Example - Link lines are to be labeled NODE-TO-NODE:

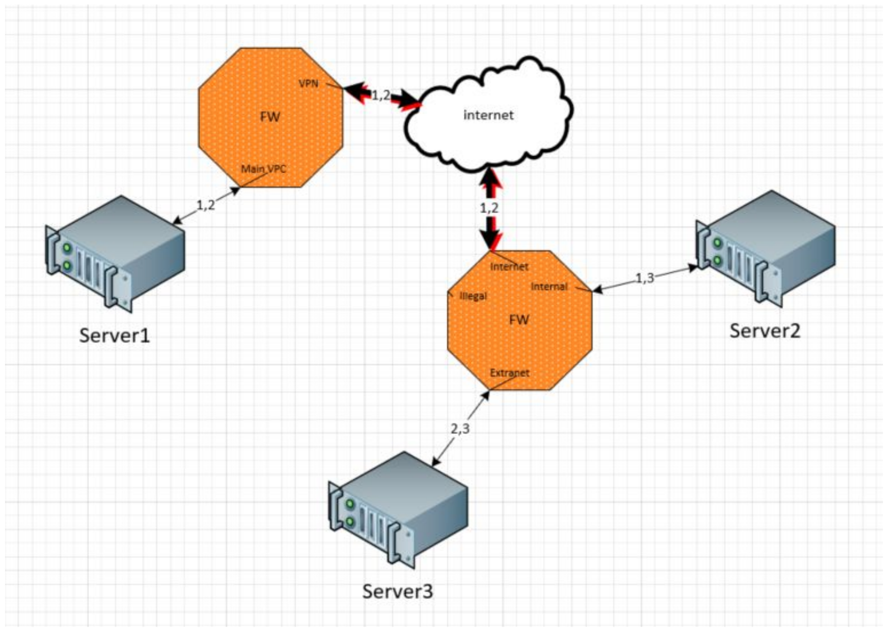
This is a really important note and concept. Always, always, always, label your links with a complete flow from ***node to node***. If the data channel passes through a device, do not waste effort relabeling it. There will be massive confusion and extra work that provides no security benefit. This applies to only node to node scenarios, do not use the same link number on two different links because they use the same protocol or perform the same work. Those are different nodes and should have different link numbers.

Bad way:



Look at this example. You can't tell which node is communicating to which other node.

Better way:



Now, you can tell exactly which servers are talking to each other and with our labeling of IPSec channels you can have that added context without creating new labels.

Challenge Diagram Checklist:

- ❏ All nodes using the same design palette.
- ❏ All nodes the same size.
- ❏ All photos eliminated from the diagram.
- ❏ All lines terminated to an object.
- ❏ All line ends visible so the use can determine if the line contains an arrowhead or not.
- ❏ All line types properly indicate the mode of transfer.
- ❏ All lines marked with at least one data flow number.
- ❏ All link labels are used between two node entities only.
- ❏ The diagram is titled on the page.
- ❏ The diagram is dated on the page.
- ❏ All relevant networks separated by the firewall are labeled on the firewall.
- ❏ All lines terminating at a firewall terminate to a labeled point on the firewall.
- ❏ All entities are labeled uniquely.