Dylan Albrecht

CIS 216

17 October 2021

**Bellingham Pipeline Case Study Project**

1. Describe the incident

Back in 1999, there was an incident where a pipeline leaked gasoline into Whatcom Creek.  The incident caused three deaths and significant damage to property and wilderness (as long as you consider Whatcom Falls Park wilderness).  Eventually, a rupture in the pipeline caused the gasoline to spill into the creek unchecked for an hour and a half, and the safety sensors built to stop this from happening did not notice the issue in time.  This caused a large amount of gasoline to spill and resulted in a fireball that traveled one and a half miles along the creek.

1. What caused the incident?

The change in pressure should have been caught by the control center, which had a number of sensors and mechanical components that were made specifically to monitor and detect pressure buildups and leaks.  Unfortunately, the pressure relief valves inside the pipeline were installed incorrectly during construction, and the company did not realize this mistake when testing the valves.  When water lines were being installed across the pipeline, a contractor for the project damaged the pipeline in a way that would lead to the initial rupture.  Some of this damage could have occurred as early as 6  years before the event took place.   This eventually led to a rupture in the pipe that spilled gasoline into the stream.  To further complicate the situation, on the day that the pipe burst the SCADA system in control of the pipeline was unresponsive.  Due to all of these factors the people in charge of the safety of the pipeline were unable to act in time, and the spilt gasoline eventually ignited and caused the disaster.

1. Summarize the cybersecurity issues present in this network

* Many of the ICSs in use in the Olympic Pipeline control room were built and designed in the 80’s and early 90’s, before systems were integrated into larger networks as a whole.
* When changes were made to the SCADA controllers, these changes were made in real time rather than being tested in a sandbox or offline system first.  This caused any errors to immediately affect the pipeline.
* Changes made on the SCADA system caused SCADA performance to degrade on the day of the incident.
* The SCADA degradation meant that data was sent to the controllers every 6 minutes rather than every couple of seconds.  This meant that system monitoring was slowed way down and mistakes could not be caught until the consequences were disastrous.
* Though it’s never been confirmed, the system administrator in charge of the SCADA equipment was likely actively programing and making changes when the rupture occurred.  This presents a number of problems.  First of all, this meant that the admin was not actually in control of the SCADA equipment.  Alerts can require immediate action at times, and if the admin was actively programming at the time it is unlikely they had full control to do what was needed in the system.  The other problem with this, which has already been stated, is that these changes are going into a live system.  There is no offline testing to catch potential errors, any changes made to the equipment will have immediate and sometimes devastating effects.
* There was no authentication for workers to access the SCADA units remotely.  Workers were able to dial-in to work in real-time on the system as long as they knew the necessary phone number.

1. Determine 3 types of attacks that would work with the vulnerabilities present in this system

* One attack that certainly could have taken place would have been a physical attack, especially by some sort of terrorist group.  Obviously this attack is always a risk and not necessarily what this question is probably trying to ask, but it would have been especially devastating in this particular system.  Because data was being sent to the SCADA controllers so infrequently (every 5-6 minutes), a well-timed physical attack would have been much more devastating than normal in these circumstances.  Granted, these attacks are always unlikely and devastating, but without being caught by the monitors and contollers it would have had even more of a powerful effect.
* A malware attack would have been effective given the fact that the SCADA updates were not tested offline.  Because there was no sandbox testing, any malware introduced in the updates would have been immediately implemented into the system.  Perhaps it would not have been caught if implemented offline, but it would have a better chance of being noticed before it did any damage.
* Any unauthorized access was possible provided an attacker knew the proper phone number to dial to get into the system.  Truth be told I’m not sure exactly what the proper name for this attack is but any malicious threat actor could have gotten in unopposed.

1. What steps should be taken to secure this network?

* The largest, most glaringly obvious change that is needed on this network is simply to apply and test any changes in an offline or sandbox scenario.  I have already stated multiple times in this paper, but any changes affecting an up and running system have the potential to cause immediate and disastrous problems.  This is far and away the biggest change that would be needed in order to secure the network.  If changes were tested in an offline system before being implemented, it’s likely any errors would have been caught beforehand.
* Proper upkeep on the SCADA systems would have prevented the system from degrading to the point where it was unresponsive.  Obviously the system became unresponsive on the worst day possible, but it should have been avoidable if more care was taken beforehand.
* Remotely accessing SCADA control units is dangerous, but not requiring any authentication is downright foolish.  Unfortunately, this happened many years ago when this was more common-place.  Using what we know today, it seems obvious that one major change needed in the network is some kind of authorization and authentication to remotely access (probably a VPN).  In addition, I would wager that any data being transferred was in plaintext prior to this event.

Works Cited

Abrams, M., & Weiss, J. (2007). *Bellingham, Washington, Control System Cyber Security Case*

*Study* (pp. 1-33, Rep.). Gaithersburg, MD: NIST.