How does Custocy detect Command & Control attacks using DGA?

During our last Sprint, Custocy's scientific team implemented a solution for the detection of malicious communications called "DGA" (Domain Generation Algorithm), whose goal is to hide "Command and Control" attacks. I'll explain what it consists of and what it changes for the security of your information system.

Let's imagine that an attacker wants to establish a connection with your computer to control it remotely, copy information, or make copies of your screen. To do this, he must establish communication between his computer and yours and start a dialogue. His computer is at a specific address on the Internet, represented by a domain name (www.evil_attacker.com).

DGA, a technique that is difficult to detect

The attacker has succeeded in placing a virus on your computer. He will now try to take control of your computer from its domain name www.evil_attacker.com. When a domain name is suspected of being malicious, it is listed, and it is easy to recognize and neutralize the attack by prohibiting any contact with this suspicious domain. The lists of malicious domains are then shared between security systems, and www.evil_attacker.com will no longer harm anyone. This is where the Domain Generation Algorithm (DGA) technique comes in. It is a program that generates thousands of domain names on the fly in a fraction of a second. The attacker uses a DGA to create a multitude of fake domain names, and among all these fake names, there is a real one that hosts his malicious program. Unfortunately, your infected computer also knows the real domain name among all the fake ones and will try to connect to the malicious domain name. To stop the attack, you have to find the malicious URL among thousands of randomly generated domain names. Moreover, the malicious domain and all the fake names can change every day. It is the modern version of the age-old problem of finding a needle in a haystack.

What if neural networks were the solution?

To give you an example, here is a fake domain name generated by a DGA called Tovar:

6fugn11ppz46e1e51xcia3brj0.org

The human eye recognizes the difference with a real domain name :

custocy.com

So wouldn't it be possible to train a machine to recognize real domain names from fake ones? Of course, it would! This has been the subject of much research over the last ten years. First of all, we started to recognize words from common language or three letters sequences commonly found in our natural language. In the end, it is neural networks (what else?) that will take the lead in the detection of false domain names and, in particular, two neural network architectures that are the memory networks (LSTM) and the convolutional networks (CNN). Without getting into too much detail, memory networks are excellent for analyzing sequences of elements such as letters or words*; they know which characters to keep in memory to distinguish the true from the false. Convolutional networks come from image analysis and are excellent for extracting groups of helpful information, in our case groups of several letters. These two types of neural networks have radically improved the recognition of domain names generated by DGA.

Problem solved?

Not quite. Unfortunately, the fantastic tool that can generate "fake" Obama videos or make you look younger in photos is also used by cybercriminals. This tool is called the adversarial neural network, and cybercriminals use it to counter our detection tools. Let's go back to our neural network that detects fake domain names. In our lab, we train our neural network on labelled data: real domain names (e.g. custocy.com) and fake domain names (6fugn11ppz46e1e51xcia3brj0.org). A neural network does not answer Yes or No. It provides a percentage of certainty about the legitimacy of each domain name analyzed. It is penalized when it is wrong and adjusts to make better predictions.

When it repeatedly returns mixed percentages, it indicates difficulty in classifying the data. Once in production, under real-world conditions, the accumulation of uncertain results over a given time tells us that there is a problem.

Now imagine that we put in front of our neural network an antagonistic neural network. This adversarial neural network will give domain names to our first network and be punished when our neural network manages to tell the difference between true and false! So we have an arms race where we try to produce neural networks that adapt faster than their malicious adversaries.

Why multiply the layers of intelligence

The Custocy solution has a hidden advantage, however. We monitor in real-time the flow exchanges in the networks we protect. Let's say we look closely at conversations between computers. But we also monitor the change in behaviour of the neural network over time. In the case of a DGA attack, a malicious conversation could escape the attention of our neural networks, which would then tell us, "I'm not sure, but I don't think this domain name is fake". The stream would then escape our surveillance. However, if, after a few thousand exchanges over a few minutes, our neural network comes up with low and uncertain scores several times in a row, our solution would be smart enough to observe this behaviour change. Another algorithm, located at a higher level of our solution, detects this drop in neural network performance— it is a sort of "chief algorithm" that immediately raises the alarm.

In short, we detect attacks thanks to our artificial intelligence algorithms, but also by observing changes in their behaviour. This dual approach allows us to filter out more threats without creating more alerts for your teams since the entire process of detection, monitoring and correction of these results are done within the Custocy Engine.

*Neural networks use numbers and not letters to train themselves, but we can easily convert letters of the alphabet into numerical values thanks to a technique called "word embedding".