|  |  |
| --- | --- |
| from: | **James Correnti** <james@shift5.io> |
| to: | innawillims@gmail.com |
| date: | Oct 23, 2020, 1:57 PM |
| subject: | Shift5 invites you to an Data Science challenge |
| signed-by: | shift5.io |
| security: | Standard encryption (TLS) [Learn more](https://support.google.com/mail?hl=en&p=tls&authuser=1) |
| : | Important according to Google magic. |

Attached are some [MIL-STD-1553B](https://en.wikipedia.org/wiki/MIL-STD-1553) data messages. This is "clean" data, ie no cyber-attacks should be here! What I am interested in seeing your approach to heuristics/training algorithms you find compelling ways to detect attacks anomalies in other traffic.

I am very flexible on how you deliver data to me (Ipython notebook over email, GitHub link, Google Drive link, whatever). If you could have something back to me by the end of next week and we will schedule a follow up to talk through your work.

Here is some useful information about 1553 and the data attached:

At a very high level, the MIL-STD-1553 bus is a data transmission protocol that consists mainly of two types of computers, Bus Controllers (BC) and Remote Terminals (RT). As indicated by the name, BCs control every action that happens on the bus, by dictating exactly when and how RTs communicate. Each 1553 message is composed of three parts. The "command word" - which is the BC message telling which RT to send/receive data, a "status word" - the response from the RT acknowledging the BC command word, and finally the 0-32 data words - the useful information being passed across the bus. All three of these "words" are 16 bits in length. The Wikipedia entry on MIL-STD-1553 is probably enough to get you started, but if you have more in-depth questions about 1553, the attached PDF is a great place to start and

Each row in the attached csv is a 1553 message (there is no assume status word, you can ignore that for this exercise). The 16-bit command word has already been translated for you, into the addr, rxtx, subaddr, and count fields. The addr field is the RT that the Bus Controller is addressing. rxtx is a boolean that indicates whether the RT being addressed is supposed to be sending or transmitting the data field. If true, the RT is sending the data, and if false the BC is sending the data. I would think of subaddr as something akin to port numbers in IP communications (each RT could have several services running on it). Count is the number of 16-bit words sent in data. Data is the actual useful information being transmitted but in raw hex.  
  
If you have any questions, please feel free to reach out - I know there is a lot to sort through here.