Cryptanalysis of HB Protocol with Machine Learning | Strahinja Praška

Introduction

- Solving Learning Parity with Noise problem, later referred as LPN, arises when trying to break HB protocol, a lightweight cryptographic protocol.
- HB protocol is mostly in card readers in hotels. We have reader(R) and tag(T), that's our RFID chip in a hotel card. If T communicated it's secret key in clear, adversary could simply sniff traffic and obtain the secret key.
- We couldn't use our usual encryption algorithms like AES in RFID chips, since it has low computational power hence the low price.
- Scheme: R is repeatedly challenging T to compute something, that something is vector a, the challenge. Now T responds back with scalar product b=<a,s>+e a and s, where s is secret key T owns, e is error term we add, we add e (flip the responding bit) with probability p. This is repeated a lot of times and is necessary for security reasons, adversary has to solve As~b which is proven to be infeasible for large enough key size. Also we calculate in GF(2) so + is actually XOR.
- We are going to threat each challenge vector a in challenge matrix A as sample and every b from vector b as label. If we throw in a basis vector we would receive a good guess for corresponding bit of key in basis vector, that's why we will use diagonal matrix when predicting the key.

Results

- Best results were given by Neural Networks by far, with max dimension learned n = 29 and 4 million samples used.
- Next-up we have modified forest algorithm I came up with that reduced sample size of single decision tree, but it wasn't that much it could guess correctly with say ~75000-100000 where single decision tree would need 100.000 samples. Meanwhile sacrificing training speed.
- Single decision tree, best I could get was n = 22 with 10 million samples, although this is fastest approach.
- Logistic regression and Bernoulli Naïve-Bayes failed to learn in general, they could learn very small dimension with bad keys which isn't useful.

Neural Network

n	Samples	**Accuracy**
15	7500	100%
16	10000	100%
17	15000	100%
18	20000	100%
19	50000	100%
20	75000	100%
21	100.000	100%
22	125.000	100%
23	150.000	100%
24	175.000	100%
25	200.000	100%
26	250.000	100%
27	500.000	100%
28	1.000.000	100%
29	4.000.000	100%
30	5.000.000	96%

Decision tree

n	Samples	**Accuracy**
15	100.000	100%
16	100.000	93.75%
17	1.000.000	100%
18	1.000.000	83.3%
19	10.000.000	100%
20	10.000.000	100%
21	10.000.000	95.24%
22	10.000.000	95.45%
23	10.000.000	65.22%
24	10.000.000	45.83%
25	10.000.000	76%
26	NA	NA
27	NA	NA
28	NA	NA
29	NA	NA
30	NA	NA

Methods

1. Neural Networks
2. Decision tree
3. Random forest
4. Bernoulli Naive-Bayes
5. Logitstic Regression

References

[1] R. Kübler, Time-Memory Trade-Offs for the Learning Parity with Noise Problem (2018)
[2] Gołębiewski, Z., Majcher, K., Zagórski, F., Zawada, M. (2011). Practical Attacks on HB and HB+ Protocols
[3] Where Machine Learning meets Cryptography (https://towardsdatascience.com/where-machine-learning-meets-cryptography-b4a23ef54c9e)