BAB V

SIMPULAN DAN SARAN

5.1 Simpulan

Dari hasil penelitian yang sudah didapatkan penulis, penulis mendapatkan beberapa kesimpulan yaitu

1. Dengan menggunakan teknologi *machine learning* pada sistem proxy mitigasi DDoS didapatkan hasil akurasi deteksi serangan DDoS dengan nilai rata rata diatas 95% ke 7 model *machine learning* yang berbeda dengan menggunakan data *features* dan metode pre-*processing* seperti normalisasi dan *balancing* SMOTE yang digunakan oleh penulis. Namun masih terjadi *overfitting* pada dataset protokol TCP akibat penggunaan dataset yang kecil sehingga tidak mewakili semua kemungkinan data acak. Semua model *machine learning* didapatkan dengan waktu deteksi dan waktu pengambil kesimpulan yang cukup kecil yaitu sekitar 0.1 – 1 detik pada perangkat laptop penulis dan 3 – 4 detik pada *virtual machine*.
2. Setiap model *machine learning* memiliki perbedaan hasil karakteristik pada setiap protokol jaringan komputer dengan matriks akurasi dan waktu prediksi, sehingga penulis menyarankan untuk memakai model *machine learning* yang berbeda beda tiap protokol untuk mendapatkan performa *machine learning* terbaik.
3. Penulis menyarankan jika sistem ini diimplementasikan pada spesifikasi sistem yang rendah, digunakan metode *machine learning* individual dan identifikasi *machine learning*. Karena secara rata-rata penggunaan CPU merupakan yang paling kecil dan efisien daripada metode yang lainnya. Namun jika diimplementasikan pada sistem yang memiliki spesifikasi lebih tinggi maka penulis menyarankan menggunakan metode *machine learning timeseries* dengan identifikasi *machine learning* karena dengan menggunakan metode individual waktu TTMnya pasti akan sekitar 0 – 5 detik karena adanya waktu tidur pada algoritmanya, dan jika metode *timeseries* akan berkisar 3-4

detik dan bisa lebih kecil jika menggunakan sistem dengan performa CPU yang lebih tinggi.

4. Untuk metode identifikasi paket yang merupakan bagian dari serangan DDoS, penulis mendapatkan kesimpulan bahwa dengan menggunakan metode pendekatan karakteristik paket terbanyak yaitu mencari nilai mayoritas sebagai *signature* paket DDoS bisa mengurangi waktu *time to mitigate* sistem mitigasi dan *throughput* sistem *reverse proxy*, namun proses komputasinya lebih besar dari pada menggunakan metode *machine learning* dan jika ada teknik serangan lainnya bisa saja tidak berfungsi dengan baik seperti terjadi pada protokol ICMP. Disisi lain, algoritma *machine learning* dapat secara efektif dalam nilai akurasi untuk melakukan identifikasi paket yang termasuk serangan DDoS dengan *features* yang digunakan dan jumlah dataset yang sedikit (400 paket data). Namun karena pengujian hanya menggunakan dataset yang kecil, maka terjadi *overfitting* akibat kekurangan semua kemungkinan data acak sehingga hasil tidak mencerminkan performa klasifikasi pada kondisi nyata. Didapatkan *machine learning* algoritma *naïve bayes* mendapatkan waktu total latih dan prediksi paling kecil dari pada algoritma lainnya. Hasil ini hanya sebagai referensi penentu algoritma machine learning yang akan digunakan nantinya.
5. Penggunaan teknologi *database* terdistribusi berbasis *blockchain* dapat meningkatkan efektivitas sistem IPS dari sisi *time to mitigate*, dengan waktu sebar dan konsensus yaitu 0,67 detik, hal ini jauh lebih cepat daripada menggunakan *blockchain* Etherium sesuai referensi penelitian terdahulu 2.1.4 yaitu 96.95 detik.
6. Pengujian kemampuan *immutability* data ketika menggunakan keypair yang salah mendapatkan hasil bahwa data yang disimpan pada *database* BigchainDB memiliki kemampuan *immutability* dengan perilaku *database* mengembalikan *error* dan transaksi tidak terbuat
7. Performa *reverse proxy* tidak cukup bagus karena dalam kasus terburuknya dapat meningkatkan *latency* sebesar 30 kali, hal ini karena penggunaan *library* netfilterQueue pada sistem *firewall controller* yang menggunakan *single*

thread walaupun arsitektur sistem *proxy controller* sudah diterapkan paradigma *multithreading*.

8. Penulis menyimpulkan bahwa sistem yang penulis bangun walaupun dengan waktu TTM dan penggunaan CPU yang lebih besar, namun sistem yang dibangun menunjukkan lebih efisien, keseluruhan, dan akurat dari sisi deteksi *false positivef* IPS / IDS Snort, karena dengan sistem yang penulis bangun bisa dengan tetap memproses paket normal selagi memblokir serangan DDoS dan tingkat *false positive* yang kecil..

5.2 Saran

Berdasarkan penelitian yang telah dilakukan oleh penulis, terdapat beberapa saran untuk penelitian ini sebagai berikut :

1. Mencoba mengurangi waktu rangkum pada metode *machine learning* individual untuk mendapatkan nilai *time to mitigate* yang lebih kecil dibandingkan waktu rangkum 5 detik, sehingga didapatkan model *machine learning* yang lebih efisien secara proses komputasi dan nilai *time to mitigate* dibanding metode *timeseries* yang mendapatkan waktu rangkum 3 – 4 detik.
2. Melakukan penambahan dan pengambilan dataset dengan melakukan implementasi pada jaringan nyata, karena dataset yang didapatkan oleh penulis pada skenario buatan bisa saja tidak mencerminkan dengan skenario aslinya. Misalnya seperti perbedaan jumlah *client* yang mengakses, jumlah *connection rate* dalam satuan waktu, jenis serangan lain, kebiasaan kondisi normal yang berbeda, dan lain lain. Dan juga dataset yang digunakan penulis terjadi *overfitting* sesuai dengan analisis hasil evaluasi *machine learning*. Sehingga bisa menguji performa model *machine learning* pada kondisi dataset jaringan nyata.
3. Menggunakan suatu model data untuk menyebarkan *signature* paket serangan DDoS selain *IP address* pada sistem distribusi data, agar dapat digunakan pada protokol TCP, UDP, dan ICMP dan menghindari masalah *IP spoofing*. Seperti menyimpan model latih *machine learning* identifikasi di penyimpan terdistribusi IPFS.

4. Meningkatkan performa dari sistem *reverse proxy* agar *latency* akses *webserver* semakin membaik, seperti melakukan optimalisasi sistem *firewall* dengan menggunakan paradigma *multithread* dalam mengecek paket data (*firewall comparator*) atau menggunakan *library* selain netfilterQueue, optimalisasi *looping*, dan memperhatikan *migration cost* penggunaan arsitektur *multithread*.