Università degli Studi di Pisa

Facoltà di scienze matematiche, fisiche e naturali

Corso di studi in Informatica

Rilevazione di anomalie di rete mediante analisi su serie temporali

Candidato: Salvatore Costantino

Relatore: Luca Deri

Anno accademico 2018-2019

1. Introduzione 3
1.1. Struttura della Tesi 4
2. Stato dell'arte 5
2.1. Signature-based intrusion detection system 5
2.2. Anomaly-based intrusion detection system 7
2.2.1. Statistical-based A-IDS 8
2.2.2. Knowledge-based A-IDS 9
2.2.3. Machine learning-based IDS 10
3. Obiettivi e motivazioni 12
4. Progettazione 14
4.1. Metriche 14
4.1.1. Metriche a breve termine 15
4.1.2. Metriche a medio-lungo termine 18
4.2. Rilevazione delle anomalie 20
4.2.1. Treshold e RSI 21
4.2.2. Prophet 25
4.3. Allarmi e mitigazione 32
4.3.1. XDP 34
4.4. Architettura software 37
5. Validazione 38
5.1. Validazione dei modelli 38
5.2. Validazione performance di rilevazione 38
6. Conclusioni e lavori futuri 39
7. Referenze 40

1. Introduzione

Il mondo odierno è sempre più interconnesso: se da un lato questo fenomeno porta innumerevoli vantaggi sociali, culturali, economici e finanziari, dall'altro si evidenziano problemi legati alla **sicurezza**, **privacy** e **gestione** degli host connessi in rete, che devono essere opportunamente gestiti; è stato infatti stimato che il costo economico dovuto ad attacchi informateci supera ampiamente i 100 miliardi di dollari annui [1]: c'è quindi una reale e concreta necessità di rilevare e mitigare in tempi brevi minacce, e più in generale, anomalie di rete. La gestione delle problematiche di rete si articola essenzialmente in due fasi:

- **rilevazione dell'anomalia**, ovvero ciò che si discosta in modo più o meno netto dalla normalità; il concetto di "normalità" a volte non è semplice da definire, e viene modellato in maniera diversa in base alle varie tecniche di rilevazione: per esempio, in questo lavoro di tesi, l'enfasi è posta sul fattore tempo;
- **gestione e mitigazione dell'anomalia**, che può essere effettuata manualmente o per mezzo di procedure informatiche automatizzate: alcuni applicativi loggano le varie attività di rete sospette, in modo da lasciare degli "indizi" ai soggetti interessati alle analisi di rete; altri agiscono in modo più deciso, cercando autonomamente di mitigare o risolvere la situazione anomala che si è verificata. Ovviamente gli approcci sopra citati possono coesistere.

Esistono varie soluzioni che gestiscono tali problemi, ognuna con i suoi pregi e difetti; in particolare in questo lavoro di Tesi viene proposta una nuova metodologia per rilevare e mitigare anomalie, che vuol essere d'aiuto ad amministratori di rete, nel prendere le opportune decisioni.

1.1. Struttura della Tesi

La Tesi è organizzata come segue:

- la **sezione 2** tratta lo stato dell'arte relativo alle soluzioni esistenti per individuare anomalie di rete;
- gli obiettivi e le motivazioni dell'applicativo prodotto vengono discusse nella **sezione 3**;
- all'interno della **sezione 4** vengono analizzate le scelte progettuali effettuate, e viene descritta l'architettura del software proposto;
- la validazione della soluzione implementata viene discussa nella **sezione 5**;
- si termina con le conclusioni e lavori futuri, presentati nella **sezione 6**.

2. Stato dell'arte

Attualmente, per rilevare anomalie di rete vengono impiegati dei sistemi chiamati *intrusion detection system* (**IDS**). Essi si possono dividere in due macro-categorie [2]:

- *signature-based IDS*: individuano anomalie basandosi su pattern comportamentali e strutturali specifici di attacchi noti. Leggere varianti di attacchi e minacce non note non vengono rilevate da questa tipologia di applicativo, quindi si ha un alto tasso di falsi negativi. D'altra parte, essi hanno un basso numero di falsi positivi [3];
- *anomaly-based IDS*: imparano il comportamento di un sistema in condizioni normali (e anormali), ed individuano un'anomalia quando il comportamento corrente differisce "troppo" da ciò che si è appreso durante la fase di training (oppure quando una certa istanza viene catalogato come anomalie, grazie agli esempi "negativi" appresi). In generale, più dati si hanno nella fase di apprendimento, migliore sarà la capacità di rilevazione. Per costruzione, sono in grado di rilevare anche minacce non note, ma sono affetti da un alto tasso di falsi positivi.

2.1. Signature-based intrusion detection system

I signature-based IDS come snort vengono configurati mediante un insieme di regole per individuare pacchetti malevoli; solitamente quando una regola "matcha" un pacchetto viene intrapresa un'azione (alert, drop ecc.). In snort, per esempio, le regole hanno la seguente forma [4]:

```
alert tcp any -> 192.168.1.0/24 70
(content:"|00 01 86|"; msg: "mountd access";)
```

dove il testo all'esterno delle parentesi costituisce la *rule header*, mentre all'interno compaiono le *rule options*; in particolare:

- all'interno dell'intestazione compaiono l'azione da intraprendere, il protocollo, gli indirizzi IP e le porte sorgente e destinazione;
- le opzioni permettono di filtrare più in profondità i vari pacchetti, e consentono di aggiungere delle informazioni di log: in questo caso si specifica la stringa da ricercare nel payload ed il testo da stampare in caso di matching.

Suricata è un ulteriore esempio di signature-based IDS, il cui formato delle regole è praticamente uguale a quello visto per snort.
Zeek (ex Bro) è un IDS che offre anche la funzionalità di rilevazione di anomalie basandosi su firma: esso non è un classico signature-based intrusion detection system, ma utilizza altre tecniche più complesse di individuazione di minacce come l'analisi comportamentale [5].
L'architettura di Zeek è costituita da due livelli principali: l'*event engine* e il *policy script interpreter*.
L'event engine trasforma il flusso di pacchetti, che riceve dal livello sottostante, in eventi ad alto livello: essi descrivono e riportano le informazioni legate ad una certa attività di rete, senza eseguire nessun tipo di analisi sull'evento generato; tale analisi (e/o la corrispondente azione) viene invece svolta dal policy script interpreter, che per ogni evento generato dal livello appena discusso, invoca l'event handler corrispondente, scritto nel linguaggio di scripting di Zeek.
In tale applicativo, un esempio firma (di ovvia semantica) è il seguente:

```
signature my-first-sig {
    ip-proto == tcp
    dst-port == 1078
    payload /.*root/
    event "Found root!"
}
```

Qualora ci fosse corrispondenza tra firma e (un) pacchetto della connessione, l'evento *signature_match* verrebbe generato:

```
    event signature_match(state: signature_state, msg: string,
                          data: string)
```

dove:

- *state* contiene informazioni più dettagliate sulla connessione che ha generato l'evento
- il contenuto di *msg* è "Found root!"
- *data* contiene l'ultima parte del payload che ha matchato con l'espressione regolare *".*root"*

A questo punto lo script che gestisce l'evento generato, effettuerà le opportune analisi ed azioni (per esempio potrebbe generare un alert).

Come si può vedere, queste tecniche di rilevazione di anomalie prevedono pattern troppo rigidi in grado di rilevare solo minacce già note: piccolissime variazioni in un attacco conosciuto o nuovi attacchi non vengono individuati da questa famiglia di IDS, poiché non ne è stata scritta ancora la corrispondente firma; il numero di falsi positivi è ovviamente molto contenuto, in quanto tali applicativi sono costruiti per individuare pattern anomali noti.

2.2. Anomaly-based intrusion detection system

I vari anomaly-based IDS usano diverse tecniche di rilevazione, ma in generale condividono le seguenti fasi o livelli [6]:

- *Parametrizzazione*: le metriche o i parametri da analizzare vengono rappresentati in una certa forma stabilita a priori, consona alle analisi da effettuare;
- *Training o allenamento*: il comportamento normale (e anormale) del sistema viene appreso e ne viene costruito il modello corrispondente;
- *Rilevazione*: il modello costruito viene confrontato con il traffico attuale alla ricerca di istanze anomalie.

Gli anomaly-based IDS possono essere suddivisi in tre tipi principali: *statistical-based* (il comportamento del sistema è visto da un punto di vista statistico), *knowledge-based* (si cerca di apprendere il comportamento desiderato, utilizzando i dati di sistema disponibili (protocolli utilizzati, traffico di rete, specifiche ecc.)) e

machine learning-based (viene costruito un modello rappresentante il comportamento normale (e talvolta anormale) delle istanze da analizzare.

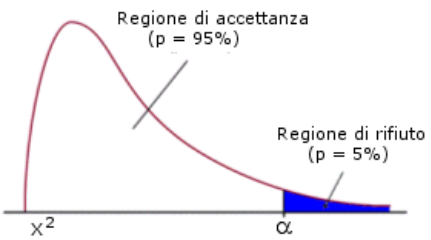
2.2.1. Statistical-based A-IDS

Gli IDS che appartengono a questa famiglia, catturano il traffico di rete e costruiscono un modello che rappresenta il suo comportamento stocastico (dipendente dal tempo). Possono essere profilate varie metriche come il traffico in entrata/uscita, flussi, pacchetti in entrata/uscita ecc.; si ha un'anomalia quando il comportamento statistico attuale si discosta "troppo" dal modello creato.
Le prime tecniche statistiche si basavano sulla costruzione di modelli ad una variabile, i cui parametri erano rappresentati da variabili aleatorie gaussiane indipendenti. A queste variabili veniva associato un valore di treshold (ovvero un valore soglia, il cui superamento rappresenta un evento eccezionale), entro il quale, il comportamento veniva considerato normale (utilizzando per esempio test statistici che si basano sulla *three-sigma-rule*). Successivamente furono proposte delle soluzioni che prevedevano la correlazione tra più variabili, che si dimostrarono essere più precisi dell'approccio illustrato precedentemente. Altre tecniche effettuano analisi su serie temporali, considerando gli eventi accaduti in un certo lasso temporale e mettendoli in relazione con ciò che si è osservato precedentemente.

Un esempio di test statistico è il test chi quadrato, che utilizza la variabile test chi-quadro, così definita [7]:

$$\chi^2 = \sum_{i=1}^{n} \frac{(\mathrm{X}_i - \mathrm{E}_i)^2}{\mathrm{E}_i}$$

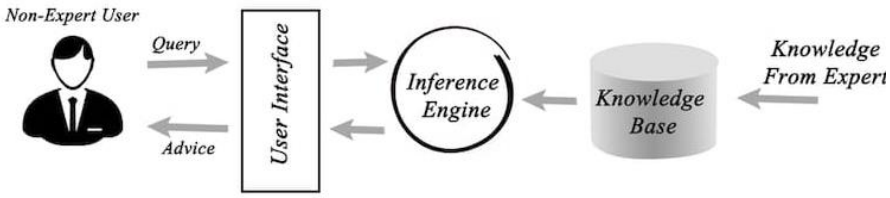
Dove *n* è il numero delle variabili da considerare, X_i è il valore dell'i-esima variabile osservate ed E_i è il suo valore atteso. Fisando l'errore tollerato e considerando le tavole della distribuzione chi-quadro è possibile stabilire se il campione di osservazioni considerato presenta delle anomalie.



Uno dei grandi vantaggi dei sistemi statistical-based, è dovuto al fatto che non richiedono alcuna conoscenza preventiva sul sistema che stanno analizzando, ma ne assimilano via via il comportamento basandosi sulle osservazioni effettuate; d'altra parte, un attaccante potrebbe generare traffico in modo da far assimilare all'IDS un comportamento anomalo, in modo che un eventuale attacco non venga rilevato. Generalmente gli applicativi appartenenti a questa famiglia di IDS possiedono una notevole capacità di rilevazione di attività anomale.

2.2.2. Knowledge-based A-IDS

Fanno parte di questi famigli i cosiddetti "expert-system". Essi classificano i dati in input secondo un insieme di regole, in tre fasi distinte: nella prima fase vengono estratti dai dati di training le classi e i relativi attributi rilevanti, ovvero gli oggetti che costituiscono il dominio d'interesse. Successivamente, dal modello costruito nella prima fase, vengono dedotte le regole di classificazione dei dati. Nell'ultima fase, in base alle procedure stabilite precedentemente, vengono classificati i dati in input da controllare.



Un'altra tipologia di knowledge-based IDS, sono i sistemi basati sulle specifiche: il modello desiderato viene costruito da esperti del settore in base a delle regole (le specifiche), cercando di catturare il normale comportamento del sistema; più la specifica risulta completa, migliore è la capacità di rilevazione. Le specifiche possono essere modellate tramite strumenti formali come gli automi a stati finite, usati soprattutto per rappresentare le attività di rete.

La robustezza e la flessibilità sono i punti di forza dei knowledge-based IDS; la costruzione di una buona base di conoscenza risulta però assi difficoltosa e dispendiosa in termini di tempo.

2.2.3. Machine learning-based IDS

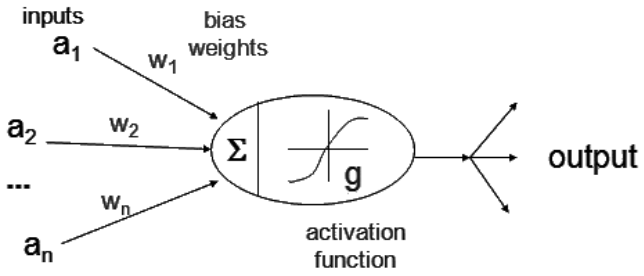
Le tecniche basate sul machine learning, si propongono come obiettivo la costruzione di modelli in grado di classificare la natura di nuove istanze che devono essere analizzate. I modelli vengono costruiti durante la cosiddetta fase di training, utilizzando i dati del training set, che solitamente sono etichettati. Esistono anche (più raramente) delle soluzioni che prevedono dati non etichettati, come mostra il lavoro svolto nella costruzione del NIDS (network IDS) Kitsune [8].

Quest sistemi hanno la capacità di migliorare la loro capacità di predizione, man mano che acquisiscono nuove informazioni. D'altra parte, gli applicativi machine-learning-based richiedono un numero considerevole di risorse spazio-temporali, soprattutto nella fase di training.

Le cosiddette *reti Bayesiane* modellizzano le relazioni probabilistiche tra le metriche d'interesse. Esse sono usate per rilevare anomalie, riuscendo a codificare le interdipendenze tra le variabili in gioco e a predire nuovi eventi. È stato dimostrato che i risultati ottenibili con reti bayesiane son comparabili con i sistemi basati su treshold, utilizzando però un numero di risorse considerevolmente maggiore; inoltre la capacità di rilevazione è dipendente dal dominio d'interesse a cui sono applicate.

Anche le catene di markov vengono usate nella rilevazione di anomalie; una catena di markov è composta da un insieme di stati, interconnessi da alcune probabilità di transizione, che determinano la topologia e la capacità del modello. Durante la fase di training vengono imparate le probabilità di transazione (da uno stato della catena ad un altro) in base al comportamento del sistema da monitorare. La rilevazione di anomalie viene effettuata confrontando lo score della sequenza di eventi osservata, con un valore di threshold fissato.

Le reti neurali, modellando il concetto di neuroni e sinapsi presenti nel cervello umano, sono molto utilizzate nell'ambito della rilevazione di intrusione, grazie alla loro flessibilità e capacità di adattamento in base alle nuove informazioni che ricevono.



Esse sono capaci di apprendere, grazie ad esempi etichettati o meno, il comportamento corretto, e talvolta anche quello anomalo, di un determinato sistema. Uno degli elementi che accomuna le varie reti neurali esistenti, è il fatto che la decisione presa non risulta umanamente interpretabile, in quanto l'apprendimento consiste nel trovare la configurazione ottimale di alcuni parametri in modo da minimizzare una certa funzione obiettivo.

anche la tecnica del clustering è usata per rilevare anomalie in un insieme di dati: essa opera raggruppando i dati osservati in alcuni agglomerati (cluster) in base alla loro reciproca somiglianza. La tecnica più comune è quella di scegliere un rappresentante di ogni cluster (centroide) e man mano che vengono osservati nuovi dati, si inseriscono nel cluster rappresentato da centroide più vicino; i punti che sono "troppo" diversi dai centroidi individuati, sono considerati anomalie. Un esempio di algoritmo di clustering è il K-NN (k-nearest neighbors), che opera partizionando i dati in base ai k punti più vicini, solitamente utilizzando la distanza euclidea.

Altre tecniche note impiegano la logica fuzzy e gli algoritmi genetici.

3. Obiettivi e motivazioni

Lo scopo di questa Tesi è la realizzazione di un applicativo, in grado di rilevare alcune problematiche di rete, efficiente in spazio e in tempo, usabile all'interno di un'intera sotto rete per monitorare gli host che ne fanno parte. Inoltre, esso deve prevedere dei meccanismi per segnale ed eventualmente mitigare le anomalie rilevate. Le problematiche di rete vengono rilevate grazie all'apprendimento del comportamento passato, a breve o a medio-lungo termine, degli host facenti parte della rete da monitorare. Il software proposto si colloca quindi nella famiglia degli anomaly-based IPS (intrusion prevention system); in particolare esso combina la semplicità e l'efficienza dell'approccio statistico (usando preliminarmente alcune soglie fisse note) nel breve termine, con la potenza e la precisione del machine learning nel medio-lungo periodo: l'applicativo quindi è un ibrido tra le famiglie statistical-based e machine learning-based discusse precedentemente. Per quanto riguarda la capacità di rilevazione, viene adottato un comportamento che intende, principalmente, minimizzare il più possibile la quantità di falsi positivi, ovvero le rilevazioni di anomalie, quando queste non sono realmente presenti: vogliamo quindi massimizzare la precisione *P*, ovvero il seguente rapporto [9]:

$$P = \frac{TP}{TP + FP}$$

dove TP rappresenta il numero dei veri positivi (rilevazione di anomalie esistenti) e FP il numero dei falsi positivi (rilevazione di anomalie non esistenti).
Altre misure statistiche della performance di rilevazione, che verranno prese severamente in esame, sono la sensitività e la specificità.

Il lavoro di Tesi intende estendere la classe degli anomaly-based IDS, costruendo un applicativo innovativo capace di monitorare il comportamento di un insieme di host in modo semplice ed efficiente; inoltre l'enfasi della rilevazione è posta, sull'analisi avanzata su serie temporali, soprattutto nel medio-lungo termine dove i concetti di trend e di stagionalità multipla risultano di fondamentale importanza.
Il fattore tempo è fondamentale per capire profondamente il comportamento di una metrica di rete, e quindi per rilevarne eventuali anomalie; per esempio, è necessario modellare esplicitamente il fatto che alcune metriche di rete presentano dei picchi fisiologici, che non sono da intendersi come anomalie: se si considera un host

situato in un ambiente lavorativo, è normale che esso non faccia alcun traffico nel weekend o dopo l'orario lavorativo, e presenti quindi un picco non appena un dipendente inizi nuovamente a lavorare.

Altre volte invece, alcune coppie di metriche, sono proporzionali tra loro, e mantengono un rapporto più o meno costante nel tempo: in tal caso è necessario innanzitutto effettuare un check preliminare sul valore del rapporto, e successivamente applicare degli indicatori di analisi tecnica, per verificare l'eventuale presenza di picchi "importanti", sintomo di probabile "malattia".

Tutti questi aspetti saranno discussi più nel dettaglio nelle sezioni successive.

4. Progettazione

In questa sezione vengono discusse, le scelte progettuali adottate in questo lavoro di tesi; in particolare viene spiegato in modo dettagliato cosa si vuole monitorare, e con quali metodologie vengono effettuate le varie analisi di rete e la mitigazione delle minacce rilevate.
Verrà inoltre illustrata l'architettura del software proposto.

4.1. Metriche

L'analisi delle metriche individuate, consentono di individuare alcuni problemi di rete comunamente riscontrati nelle reti odierne; dove necessario, sono state messe in correlazione coppie di metriche in modo da poter effettuare analisi in modo preciso e coerente.

Sono state individuate due macro-famiglie di metriche:

- Metriche a breve termine: esse sono essenzialmente formate da coppie di metriche correlate tra loro: data la coppia di metriche (x, y), viene considerato il rapporto x/y; ipotizziamo in modo ragionevole, che numeratore e denumeratore siano "più o meno" proporzionali tra loro, quindi importanti disproporzionalità verranno considerate anomali: inoltre è possibile impostare dei valori di treshold sul rapporto precedentemente descritto;
- Metriche a medio-lungo termine: la loro caratteristica principale è data dal fatto che esse presentano trend e delle stagionalità dipendenti dal tempo; Queste metriche verranno analizzate da un algoritmo che sfrutta in modo cruciale il loro comportamento temporale.

Precisiamo ulteriormente che entrambe le famiglie di metriche dipendono dal tempo di osservazione, in altre parole data la metrica (o coppia di metriche) x, siamo interessati alla funzione $f_x(t)$ che ne esprime il comportamento in funzione del parametro t (tempo).

Tutte le metriche sono state scelte in seguito ad una lunga ed attenta analisi delle loro proprietà: tra le innumerevoli metriche esistenti, se ne sono scelte alcune tra le più rilevanti ai fini delle analisi che si vogliono effettuare; i valori delle metriche sono stati ottenuti utilizzando il software di monitoraggio di rete ntop. Alcune delle metriche (o meglio, i contatori relative ad esse e le corrispondenti serie temporali) erano già presenti in ntop, altre sono state implementate appositamente per questo lavoro di tesi; infatti, sono state create le serie temporali relative a flussi host unreachable, pacchetti ARP, statistiche e pacchetti TCP, pacchetti ICMP echo, e pacchetti DNS.

4.1.1. Metriche a breve termine

Le metriche a breve termine vengono analizzate dal sistema ogni cinque minuti, alla ricerca di picchi o valori anomali; inoltre è necessario un periodo di almeno un'ora, per inizializzare il comportamento delle metriche di rete considerate, al fine della rilevazione di eventuali pendenze sospette. Come detto in precedenza, vengono analizzate coppie di metriche, opportunamente correlate.

In particolare, data la coppia di contatori (x, y) *relativi alle metriche* $(M_{x,}\ M_y)$ *da analizzare, consideriamo il loro rapporto ed eventualmente le variazioni di esso:*

$$\frac{\Delta x}{\Delta y} = \frac{x_{t_f} - x_{t_i}}{y_{t_f} - y_{t_i}} = r_{t_i}, \qquad \Delta r = r_{t_f} - r_{t_i}$$

Le tecniche usate per analizzare tali informazioni, verranno discusse in seguite.

Le metriche a breve termine (o meglio, le coppie di metriche) che vengono prese in esame sono le seguenti (rispettivamente nel ruolo di numeratore e denominatore):

- **Risposte DNS ricevute (inclusi gli errori) e query DNS inviate**: poiché in situazioni normali, ad ogni query del protocollo di risoluzione degli indirizzi corrisponde una e una sola risposta, idealmente il loro rapporto deve essere uguale ad 1, o minore di 1 ma molto vicino ad esso, a causa della non affidabilità del protocollo UDP; valori maggiori di 1 indicano sicuramente la presenza di un problema che va investigato: potrebbe essere dovuto semplicemente ad una cattiva configurazione di rete, o nel peggiore dei casi ad attacchi di tipo flooding [10].

- **Risposte DNS con errore ricevute e risposte DNS corrette ricevute**: ricevere occasionalmente riposte DNS con codice d'errore può essere un fatto fisiologico, ma è necessario confrontarle con il numero di risposte corrette: un'elevata percentuale di errori rispetto alle risposte corrette (un rapporto uguale al 50%, risulta già essere sospetto) può indicare un'errata impostazione di rete o problematiche relative al name server o resolver, che devono essere risolte in tempi brevi.
- **Bytes protocollo DNS inviati e pacchetti DNS inviati**: tale rapporto esprime la dimensione media dei pacchetti DNS inviati (query ed eventuali risposte inviate in qualità di name server); a causa della non affidabilità del protocollo UDP, per evitare frammentazioni del pacchetto in transito, si mantiene la dimensione del pacchetto UDP (compreso il payload) minore di 576 bytes, che è il minimo valore del MTU (maximum transmission unit). Quindi, se la dimensione media dei pacchetti DNS è maggiore di 576 byes, potrebbe essere sintomo di un errata configurazione del resolver o name server, oppure nel caso peggiore potrebbe trattarsi di "data exfiltration" [11] o di un attacco DNS flooding di tipo amplificazione.
- **Bytes protocollo DNS ricevuti e pacchetti DNS ricevuti**: valgono tutte le considerazioni fatte nel punto precedente, con la differenza che in questo caso il problema può essere l'infiltrazione di dati o l'essere vittima di un attacco DNS flooding.
- **Risposte ICMP echo e richieste ICMP echo:** I pacchetti ICMP di tipo echo vengono comunamente inviati per verificare lo stato d'attività di un host connesso alla rete; idealmente il rapporto risposte e richieste è uguale a 1, ma potrebbe anche essere minore di 1 (ma comunque molto vicino ad esso) a causa di eventuali richieste perse; valori maggiori di 1 indicano certamente un problema, che va dall'errata configurazione di rete ad attacchi di tipo flooding.
- **Flussi ICMP port unreachable come client e Flussi Totali come server:** Un flusso in uscita di tipo port unreachable indica che un qualche host ha provato ad accedere ad un servizio non presente sulla macchina di destinazione; ricevere o inviare occasionalmente pacchetti ICMP port unreachable è del tutto normale, ma è necessario rapportarli al numero di

flussi totali nel periodo considerato: se ci sono troppi flussi ICMP port unreachable rispetto ai flussi totali, oppure si verifica un picco positivo importate rispetto alla variazione di due rapporti consecutivi, allora è necessario investigare la situazione in quanto potrebbe trattarsi di port scan (come vittima, poiché stiamo considerano i port unreachable inviati).

- **Flussi ICMP port unreachable come server e Flussi Totali come client:** valgono gli stessi ragionamenti espressi nel punto precedente, ma in questo caso un'anomalia legata a tale rapporto potrebbe essere sintomo di port scan effettuato, quindi l'host anomalo verrà catalogato come attaccante.
- **Flussi ICMP host unreachable come client e Flussi Totali come server:** Un flusso host unreachable in uscita viene generato dall'invio di un pacchetto ICMP host unreachable da parte di un host (solitamente un router) che non riesce ad inoltrare sulle sue porte di uscita un qualche pacchetto in arrivo sulle sue porte di input, poiché la sua tabella d'inoltro non presenta una rotta per un certo indirizzo IP; occasionalmente è fisiologico, inviare o ricevere questo tipo di pacchetti, ma è necessario metterli in relazione con il totale dei flussi ed individuare eventuali picchi nel rapporto inizialmente considerato; anomalie su questa metrica possono indicare la presenza di attività malevola di un worm (che sta attaccando la rete dell'host analizzando), il quale nelle fasi iniziali provvede a ricercare indirizzi IP (generati spesso casualmente) di host con vulnerabilità da sfruttare.
- **Flussi ICMP host unreachable come Server e Flussi totali come client:** si considerano le analisi fatte nel punto precedente, con la differenza che l'host anomalo si configura come attaccante.
- **Richieste ARP inviate e risposte ARP ricevute**: idealmente ad ogni richiesta ARP corrisponde una e una solo risposta ARP: qualora il rapporto fosse troppo sbilanciato a favore delle richieste ARP, o presenti picchi importanti allora è possibile che l'host analizzato stia attuando un network discovery; non è ovviamente un problema grave, ma potrebbe essere un'informazione utile ad un amministratore di rete.
- **Pacchetti TCP ritrasmessi, persi, out of order e pacchetti TCP totali:** pacchetti TCP persi, ritrasmessi e out of order, sono comuni durante una comunicazione TCP; il rapporto con il totale dei pacchetti TCP deve però

mantenersi più o meno costante, e non deve essere troppo alto, altrimenti ciò può indicare la presenza di alcuni problemi di rete, come la congestione.

- **Flussi anomali e Flussi Totali:** un flusso è considerato anomalo quando presenta delle caratteristiche particolari come la lunga durata, l'elevato volume, host partecipanti presenti in blacklist eccetera; se tali flussi occorrono occasionalmente, non devono destare particolari sospetti; se invece se ne hanno troppi rispetto al totale dei flussi, o abbiamo un picco importante considerando la variazione tra rapporti, allora ciò potrebbe indicare una probabile attività malevola in atto.

4.1.2. Metriche a medio-lungo termine

Le metriche a medio-lunghe termine, a differenza delle metriche a breve termine, sono costituite da una singola componente (non coppie di componenti), e non sono quindi presenti correlazioni tra metriche, seppure possono essere corredate da controlli che coinvolgono altre metriche, come sarà spiegato in seguito.
Tali metriche vengono controllate dal sistema ogni ora, alla ricerca di comportamenti inattesi; inoltre i valori delle metriche devono preventivamente essere osservati per un periodo minimo di tre settimane, per poter poi rilevare eventuali anomalie. In particolare, esse devono possedere (o meglio, supponiamo che posseggano) delle caratteristiche ben precise, ovvero devono avere un trend e delle stagionalità in funzione del tempo: per esempio, i valori associati ad una certa metrica crescono, decrescono o oscillano in un certo periodo temporale, ed il comportamento generale della metrica stessa tende a ripetersi nel tempo, individuando quindi stagionalità ed eventuali sub-stagionalità. Ovviamente le caratteristiche di tali metriche, ci guidano nella scelta di un algoritmo che sfrutti le proprietà sopra descritte.

Le metriche a medio-lungo termine, analizzate in questa Tesi, sono le seguenti:

- **Bytes inviati**: i bytes che sono stati inviati, ovvero il traffico in uscita di un host;
- **Bytes ricevuti**: i bytes che sono stati ricevuti, ovvero il traffico in entrata di un host;

- **Flussi come Client**: rappresentano i flussi inizializzati dall'host analizzato;
- **Flussi come Server**: rappresentano i flussi inizializzati da host che iniziano la comunicazione con la macchina sotto analisi;

tutte queste metriche condividono le seguenti osservazioni: se ci si discosta troppo dal consueto trend e stagionalità della metrica sotto esame, ciò potrebbe essere dovuto a problematiche di rete più o meno gravi: l'host analizzato ha presentato un cambiamento di comportamento importante, per una qualche ragione, che va investigata dall'amministratore di rete; per esempio l'host analizzato potrebbe essere diventato un flooder, o essere vittima di un attacco flooding.

Come accennato inizialmente, è possibile corredare la rilevazione di anomalie, con ulteriori controlli che considerano la presenza dei seguenti protocolli o eventi:

- **Protocolli sconosciuti**: protocolli la cui natura non è nota, ovvero non corrisponde a nessun protocollo noto;
- **Protocolli di accesso remoto:** molti di questi protocolli, posso essere presi di mira da malintenzionati per prendere il controllo di host remoti;
- **Comunicazione con host, catalogati come malware:** si tratta di host inseriti in delle apposite blacklist, catalogati come malware;
- **Comunicazione con host, catalogati come responsabili di attività di mining:** come nel punto precedente, si tratta di host presenti in delle blacklist.

Quindi, nel caso in cui venga rilevata un'anomalia considerando la prima lista di metriche (bytes inviati, bytes ricevuti, etc.), si può eventualmente controllare che nello stesso periodo temporale ci sia stato o meno traffico in entrata o in uscita (in base alla metrica anomala), relativo alle categorie di protocolli elencati sopra: in questo modo la precisione di rilevazione migliora notevolmente, riducendo il numero di eventuali falsi positivi fisiologici, dovuti ad un cambio di comportamento benigno dell' host sotto analisi.

4.2. Rilevazione delle anomalie

Rilevare un'anomalia significa individuare eventi che per qualche loro caratteristica non sono considerati essere normali; Ai fini delle analisi che vengono effettuate è essenziale, quindi, definire il concetto di normalità: senza definire ciò che è considerato normale, è impossibile definire ciò che è anomalo. Questa semplice considerazione è il punto di partenza per tutti i ragionamenti effettuati successivamente.

Nella fattispecie, ci preoccupiamo di definire quali sono i parametri di normalità delle metriche di rete che ci apprestiamo ad analizzare; per far ciò è importante capire la natura delle metriche stesse.

Alcune volte è facile individuare anomalie di alcune metriche, soprattutto quando esse non sono analizzate come entità a sé stanti, ma vengono correlate ad altre metriche; per esempio consideriamo le query DNS inviate e le risposte DNS ricevute da host: se il rapporto tra risposte e richieste risultasse maggiore di 1, ci sarebbe certamente un qualche problema; in tal caso è immediato utilizzare banali tecniche a soglia fissa, per rilevare l'anomalia.

In altri casi è certamente più complesso, e le metodologie utilizzate sono maggiormente soggette al problema dei falsi positivi; per esempio considerando come prima una coppia di metriche, risposte DNS con errori e risposte DNS corrette, possiamo preliminarmente, applicare delle tecniche a soglia fissa, e se queste danno esito negativo (ovvero assenza di anomalie), utilizzare delle tecniche che rilevino picchi nel rapporto delle metriche sopra considerate: assumiamo quindi in modo ragionevole che il rapporto tra risposte con errori e senza errori debba essere più o meno costante, e qualora si rilevi una grande disproporzionalità è importante analizzare attentamente la situazione verificatosi.

Altre volte non è possibile correlare più metriche, al fine della rilevazione di anomalie di rete come, nel caso dei bytes inviati ovvero il traffico totale in uscita: in tal caso una possibile idea potrebbe essere quella di correlare tale metrica con i bytes ricevuti e assumere una certa proporzionalità tra loro; ma ciò nello scenario moderno, è un'assunzione poco veritiera, in quanto in base al servizio richiesto il rapporto tra le due metriche può variare senza essere sintomo di attività malevola.

In tal caso è necessario apprendere il comportamento (passato) della singola metrica, e se in futuro il comportamento della metrica dovesse mutare profondamente rispetto al passato, allora potrebbe esserci un'anomalia.

In estrema sintesi, il comportamento normale delle metriche può essere definito da:

- Non superamento dei valori di treshold
- Assenza di picchi ripidi
- Comportamento futuro coerente con quello passato

In particolare, i primi due punti definiscono il comportamento normale delle metriche a breve termine, mentre l'ultimo punto quello delle metriche a medio-lungo termine.

Tutti gli aspetti accennati in tale sezione, verranno approfonditi nelle successive sottosezioni.

4.2.1. Treshold e RSI

Le tecniche che verranno discusse si applicano unicamente alle metriche a breve termine, ovvero quelle che vengono analizzate dal sistema ogni cinque minuti.

I sistemi a soglia (treshold), vengono utilizzati per individuare con ottima precisione anomalie nelle metriche (come spiegato più volte, si tratta in realtà di coppie di metriche) di rete; in particolare, sono state individuate tre tipi di soglia, che si applicano a diverse tipologie di metriche:

- **Soglia fissata al valore di 1.1**, per le seguenti metriche: rapporto risposte e query DNS, rapporto richieste e risposte ICMP echo e rapporto richieste e risposte ARP; idealmente, valori maggiori di uno di tali rapporti, in base alle caratteristiche dei protocolli considerati (DNS, ICMP, ARP), indicano la presenza di un problema; Si noti che la soglia è stata fissata ad un valore leggermente maggiore di 1, per evitare falsi positivi dovuti al possibile rumore presente nei dati raccolti e successivamente analizzati.
- **Soglia fissata al valore di 550,** per le seguenti metriche: rapporto bytes DNS inviati e pacchetti DNS inviati e rapporto bytes DNS ricevuti e pacchetti DNS ricevuti; la soglia quindi limita la dimensione media dei pacchetti DNS inviati e ricevuti, individuando quindi eventuali fughe ed

infiltrazioni di dati usando il protocollo DNS in modo improprio. Siccome il protocollo DNS si poggia sul protocollo di livello trasporto UDP, per evitare la frammentazione del pacchetto, solitamente si mantiene la dimensione del payload DNS minore o uguale a 512 bytes in modo da rientrare nel valore minimo assunto dall' MTU (576 bytes), ed evitare quindi la frammentazione del pacchetto. Anche in questo caso si mantiene la soglia su un valore leggermente maggiore di 524 bytes, in modo da minimizzare l'impatto del rumore presente nei dati analizzati.

- **Soglia fissata al valore di 0.85 per le altre metriche a breve termine**: si tratta sempre dei consueti rapporti tra metriche, ed un eventuale valore maggiore di 0.85 indica chiaramente un'anomalia che necessita di ulteriori controlli; per esempio se il rapporto tra risposte con errore e risposte senza errori del protocollo DNS supera il valore imposto dalla soglia, si tratta chiaramente di un problema in quanto, per ogni risposta DNS corretta ne corrisponde quasi un'altra errata. Analoghi ragionamenti valgono per le altre metriche a breve termine, non citate nei due punti precedenti.

I valori di treshold riescono quindi a identificare in modo abbastanza preciso un'anomalia; per alcune metriche, quando non viene superato il valore di soglia, ci si trova in una sorta di zona grigia, in cui è difficile stabilire se la metrica sia o meno anomala. In tal caso è necessario controllare la presenza di picchi nella variazione dei rapporti tra due periodi di tempo consecutivi, come spiegato nella sezione delle metriche a breve termine.

Per far ciò, viene utilizzato un indicatore statistico chiamato **RSI** (Relative Strength Index)[12]; esso è uno degli oscillatori più popolari dell'analisi tecnica, ovvero dello studio dei prezzi dei mercati finanziari.
Esso è chiamato oscillatore poiché varia tra due valori ovvero tra 0 e 100; siccome in ambito finanziario è usato per valutare la velocità del movimento dei prezzi, sfruttiamo questa sua caratteristica per analizzare la velocità con cui cambiano i valori associati ad una metrica di rete.

Si noti la capacità dell'RSI nel rilevare la presenza di picchi, sul seguente grafico rappresentate il prezzo di un titolo azionario:

È possibile fissare un valore limite entro il quale il valore dell'RSI calcolato deve sottostare, per far si che la metrica che stiamo analizzando non presenti picchi troppo ripidi; tale caratteristica, fa dell'RSI il candidato ideale, per la rilevazione di picchi su serie temporali. Nei mercati finanziari infatti si considerano normali, valori dell'RSI che oscillano tra 30 e 70 (nel grafico precedente, tali limiti sono rappresentati dalle due barre orizzontali della funzione in blu). Per le analisi sulle metriche a breve termine, non viene preso in considerazione alcun limite inferiore al valore dell'RSI, in quanto, per come sono state definite le metriche, un trend negativo non indica la presenza di anomalie.

Come detto precedentemente, è necessario definire un limite superiore, per rilevare le metriche che crescono troppo velocemente: è stato deciso di optare per un valore di threshold pari a 0.85, in quanto ciò garantisce un buon trade-off tra falsi positivi e falsi negativi; quest'ultimo aspetto verrà approfondito nella sezione riguardante la validazione dei modelli.

Un ulteriore parametro da scegliere per calcolare l'RSI è il numero dei periodi, ovvero il numero di variazioni dei rapporti consecutivi temporalmente, che vogliamo considerare: Se si sceglie di analizzare x periodi, allora ci serviranno $x + 1$ punti (ovvero rapporti) per poter calcolare le x variazioni richieste. In questo lavoro di Tesi si considerano 50 periodi, per ragioni che saranno spiegate più avanti nella sezione di validazione.

La formula per il calcolo dell'RSI è la seguente:

$$RSI = 100 * U/(U + D)$$

Dove U è la media delle differenze positive tra punti consecutivi nel periodo considerato, e D è la media delle differenze negative tra punti consecutivi nel periodo considerato. Sia N il numero di periodi, ed x , un array di dimensione $N + 1$ contenente i valori della metrica da analizzare, temporalmente consecutivi, allora in formule U e D si possono esprimere come:

$$U = \frac{\sum_{t=2}^{N+1} \max(0,\ x_t\ -\ x_{t-1})}{N}$$

$$D = \frac{\sum_{t=2}^{N+1} |\min(0,\ x_t\ -\ x_{t-1})|}{N}$$

Si noti, che per ogni metrica a breve termine di ogni host analizzato è necessario mantenere in memoria N + 1 punti, in modo che all'arrivo del prossimo punto venga effettuato uno shift verso destra degli N + 1 punti precedenti e venga inserito il nuovo punto per ricalcolare il valore dell'RSI. Possiamo bypassare tale inconveniente calcolando la formula precedente per i primi N + 1 punti, e per i successivi valori si procede approssimando il valore RSI esatto, calcolando U e D con la seguente formula:

$$U = (U_{old} * (N - 1) + \max(0, x_{new} - x_{last}))/N$$

$$D = (D_{old} * (N - 1) + |\min(0, x_{new} - x_{last})|)/N$$

Si noti che tale approssimazione è simile a quella effettuata nel calcolo dalla media mobile esponenziale.

Il calcolo dell'RSI (per host) è ovviamente molto efficiente in tempo, in quanto sono necessarie poche operazione aritmetiche di base: nel calcolo del primo valore dell'RSI sono necessarie N addizioni e due operazioni moltiplicative, mentre successivamente, per calcolare i valori approssimati, bisogna effettuare quattro operazioni moltiplicative e una somma.

Il costo in tempo, è pari al numero di punti necessari per calcolare il primo valore dell'RSI, ovvero N + 1.

Per evitare che l'RSI includa nel calcolo, valori palesemente anomali, che andrebbero ad inquinare le future rilevazioni e lo renderebbero inutilizzabile nel

rilevare picchi importanti se il comportamento anomalo persiste, si procede ad eseguire il controllo preliminare con la tecnica delle soglie fisse: se un valore è palesemente anomalo, questo non verrà usato nel calcolo dell'RSI, per le motivazioni espresse sopra;

Bisogna fare ulteriori restrizioni sul tipo di valori che contribuiranno al calcolo dell'RSI: Siccome si vuole rilevare il comportamento di una metrica, vogliamo che i valori registrati individuino precisamente l'andamento della metrica stessa, e ciò si può ottenere considerando i dati raccolti in condizioni sufficienti di volume del traffico di rete; tali condizioni minime di significatività dei valori registrati, dipendono dalla metrica in esame; per esempio per la rilevazione di flussi anomali si sono stabiliti le seguenti condizioni di traffico minimo: 2 flussi con comportamento sospetto oppure 12 flussi totali; al di sotto di questi valori si ritiene che il traffico non sia sufficiente a delineare il reale comportamento della metrica in questione.

I risultati sulla capacità di predizione delle tecniche del treshold e dell'RSI, saranno accuratamente discusse nella sezione riguardante la validazione.

4.2.2. Prophet

La scelta di un algoritmo in grado di analizzare metriche a medio-lungo termine è avvenuta attraverso una lunga e attenta analisi delle varie alternative esistenti; Inizialmente si è considerato il modello ARIMA (Autoregressive Integrated Moving Average) [13]; esso è un modello statistico relativamente semplice e veloce, ma non è in grado di gestire precisamente alcuna stagionalità (che è invece una caratteristica delle metriche che si vogliono analizzare) ed i suoi parametri sono difficili da scegliere a priori senza un processo di fitting.
Successivamente si è analizzato il modello HW (Holt-Winters) [14]. L'HW, come L'ARIMA, è un modello statistico semplice ed efficiente in grado di effettuare predizioni sui valori futuri assunti da una metrica. I suoi parametri possono essere scelti mediante tecniche di training; a differenza dell'ARIMA, esso riesce a catturare la stagionalità della metrica; il problema di questo metodo è la modellizzazione una singola stagionalità, e come detto in precedenza le metriche che si vogliono analizzare presentano stagionalità a periodicità multiple.

Si è quindi cercato qualcosa di più espressivo, in grado di catturare le caratteristiche delle metriche da analizzare: è stata studiata e testata la rete neurale ricorrente LSTM (Long short-term memory) [15]. Essa si è rilevata in grado di carpire le multi-stagionalità presenti nelle metriche, grazie ad un potente meccanismo di mapping dei valori della serie temporale, presente al suo interno. Il suo costo in spazio e in tempo si è rilevato però troppo oneroso per poter essere impiegato in una rete con centinaia di host da monitorare.
Tutte le tecniche illustrate precedentemente consentono di predire nuovi valori della serie temporale analizzate, ma non incorporano nessun meccanismo che consenta di rilevare eventuali anomalie.
Successivamente la ricerca si è concentrata su modelli con un potere di espressività e costo computazionale intermedio tra i modelli statistici ARIMA e HW, e il modello di deep learning rappresentato dalle reti LSTM: il sistema scelto si chiama Prophet, e le sue peculiarità vengono sotto discusse.

Le metriche a medio-lungo termine vengono quindi analizzate utilizzando Prophet, un sistema messo a punto dagli sviluppatori di Facebook. Esso consente la creazione di un modello di **regressione**, mediante l'utilizzo di parametri umanamente interpretabili e modificabili da esperti del dominio delle serie temporali da trattare [16]. In breve, Prophet consente la predizione dei valori futuri associati ad una certa metrica, mediante l'apprendimento del comportamento passato della metrica stessa.

In particolare, Prophet utilizza un modello decomponibile in tre sottocomponenti principali: trend, stagionalità e festività, combinati in una delle due forme seguenti:

$$y(t) = g(t) + s(t) + h(t) + \varepsilon_t \ (additivo)$$

$$y(t) = g(t) * s(t) + h(t) + \varepsilon_t \ (moltiplicativo)$$

dove:

- $g(t)$: è la funzione *trend*, che modella cambiamenti non periodici dei valori della serie temporale;
- $s(t)$: è la funzione *stagionalità*, che modella i cambiamenti periodici (giornalieri, settimanali, annuali, ecc.) della serie temporale;

- $h(t)$: è la funzione *festività,* rappresentante l'effetto delle festività che occorrono in alcuni giorni ben precisi; se una festività si ripete nel tempo, Prophet riuscirà a modellare in modo più preciso il comportamento della metrica analizzata nei giorni di festa specificata; altrimenti, se essa non si ripete, essa verrà modellata ma non verrà inclusa nelle predizioni future
- ε_t: è il termine rappresentante l'errore, derivante da eventuali cambiamenti nella serie temporale non previsti dal modello; per ipotesi essa viene considerata normalmente distribuita.

In Particolare, la funzione *trend,* $g(t)$, solitamente è una componente lineare mentre la funzione *stagionalità,* $s(t)$, si basa sulla serie di Fourier e assume la seguente forma:

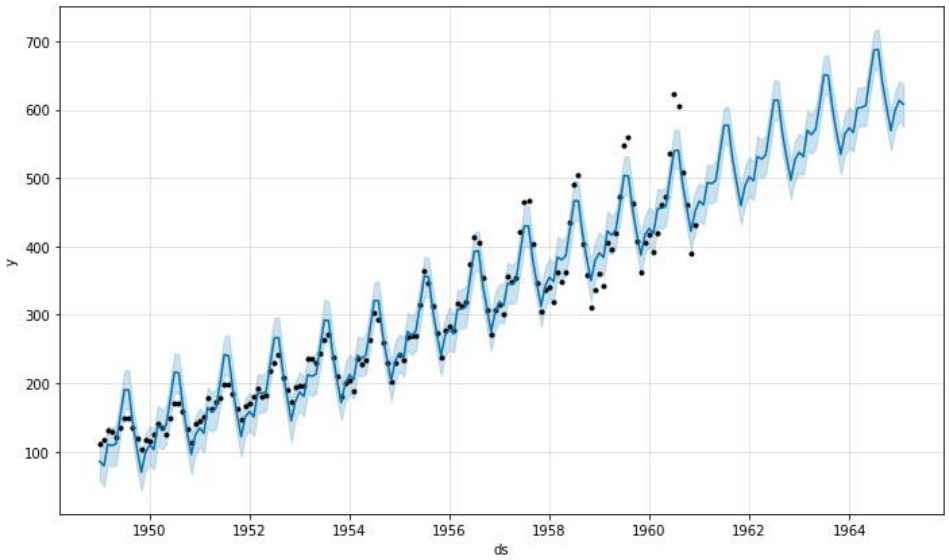
$$s(t) = \sum_{n=1}^{N} (a_n \cos(2\pi nt/P) + b_n \sin(2\pi nt/P))$$

dove, N sono i termini della serie di Fourier, P è il periodo considerato (7 giorni per periodicità settimanale, 1 giorno per periodicità giornaliera, ecc.) e i coefficienti $a_1, a_2, \dots, a_n, b_1, b_2, \dots, b_n$, sono $2N$ parametri che dovranno essere fittati per approssimare la stagionalità considerata.
Si noti che aumentando il valore dell' iper-parametro N, il modello rischia di andare in overfitting, con l'effetto di avere una limitata capacità di generalizzazione e quindi di predizione di nuovi valori.

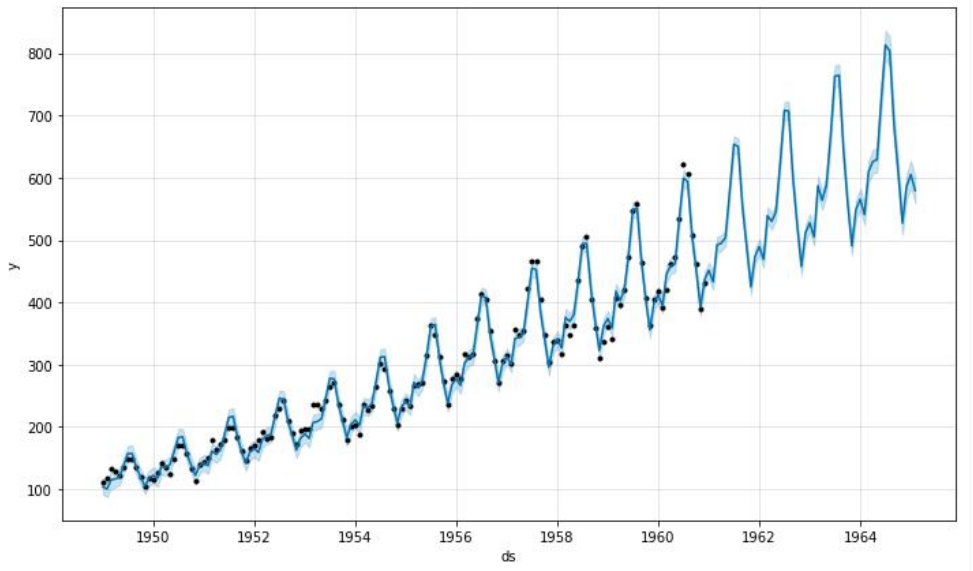
Il modello creato viene fittato alla curva rappresentante la serie temporale reale, tramite la metodologia L-BFGS [17].

Come, accennato all'inizio di questa sezione, con Prophet è possibile definire due tipi di modello, quello moltiplicativo e quello additivo; di default Prophet utilizza il modello additivo dove la componente stagionale viene sommata al trend per effettuare la predizione [18]; tale modello non funziona quando la stagionalità cresce con il trend, ed in tal caso è necessario adottare un modello moltiplicativo. Si notino a proposito i due grafici sottostanti, il primo generato con un modello additivo, il secondo con la variante moltiplicativa; il modello additivo non riesce a fittare correttamente la curva della funzione da approssimare, poiché è evidente che la stagionalità si amplifica, al crescere del trend. In tal caso è necessario scegliere

un modello moltiplicativo. Per le metriche a medio-lungo termine da monitorare si è optato per un modello moltiplicativo, poiché si è rilevato più preciso rispetto a quello additivo; tale considerazione verrà giustificata nella sezione riguardante la validazione dei modelli (*Sezione 5.1*).



modello additivo



modello moltiplicativo

Gli iper-parametri che vengono presi in considerazione per la costruzione del modello adatto agli scopi di questa tesi sono:

- **changepoint_prior_scale**: definisce la quantità di punti, in corrispondenza dei quali si verifica un cambiamento di tendenza; un valore elevato di questo iper-parametro può portare il modello ad una situazione di overfitting;
- **seasonality_prior_scale**: tramite questo parametro è possibile controllare la capacità con cui la componente stagionale riesce a fittare i dati; aumentandone il valore, il modello può andare in overfitting;
- **ordine serie di Fourier:** definisce il numero di termini della serie di Fourier; ordini maggiori consentono una migliore approssimazione della funzione da fittare, mediante l'uso di funzioni sinusoidali, e ciò può portare al fenomeno dell'overfitting.

I tre precedenti parametri vengono scelti, mediante una procedura di **model selection**, di cui si parlerà nella sezione riguardante la validazione.

Si sono individuati due tipi di stagionalità presenti intrinsecamente (per ipotesi) nelle metriche che si vogliano analizzare con questo metodo, che ricordiamo essere i bytes ricevuti/inviati e i flussi come server/client: consideriamo periodicità settimanale e giornaliera; in particolare la periodicità giornaliera è suddivisa in due sub-stagionalità, ovvero periodicità dei giorni feriali e periodicità del weekend (sabato e domenica).
Come accennato precedentemente, si ipotizza che le metriche analizzate abbiano le periodicità individuate, ma ovviamente ciò non sempre è vero; l'ipotesi fatta non è però così distante dalla realtà: si pensi, per esempio, agli host presenti in un laboratorio scolastico; ci si può aspettare che sabato e domenica il traffico sia assente e nei giorni settimanali ci sia traffico solo durante l'orario delle lezioni: in tale scenario si può notare la presenza delle stagionalità feriali e festive (sabato e domenica), e di conseguenza della stagionalità settimanale; inoltre, impostando i giorni del periodo delle vacanze scolastiche come giorni festivi, è possibile predire l'assenza di traffico nel periodo considerato.
Concludendo, quando le stagionalità individuate esistono realmente nelle metriche analizzate allora Prophet esprimerà la sua massima capacità di predizione; se invece le stagionalità non sono chiaramante visibili, allora il modello creato avrà una minore capacità di predizione.

L'analisi delle metriche di rete a medio-lungo termine, si divide in più fasi, ben distinte. Innanzitutto, è importante precisare che viene costruito un modello per

ogni metrica di ogni host, poiché ogni metrica ha un comportamento unico, così come gli host.

La prima fase consiste nel recuperare i dati da analizzare dal database contente le serie temporali.

Successivamente si procede a verificare che ci siano abbastanza punti di una certa metrica, per poter eseguire il training e la conseguente predizione; lo scenario ideale sarebbe avere tre settimane di dati completi (senza punti mancanti nella serie temporale); poiché Prophet gestisce bene anche serie temporali incomplete, ci si accontenta di avere almeno il 70% dei punti di tre settimane di dati completi, per potere iniziare con l'analisi della metrica. Ovviamente, più punti si hanno a disposizione, migliore sarà la capacità di predizione.

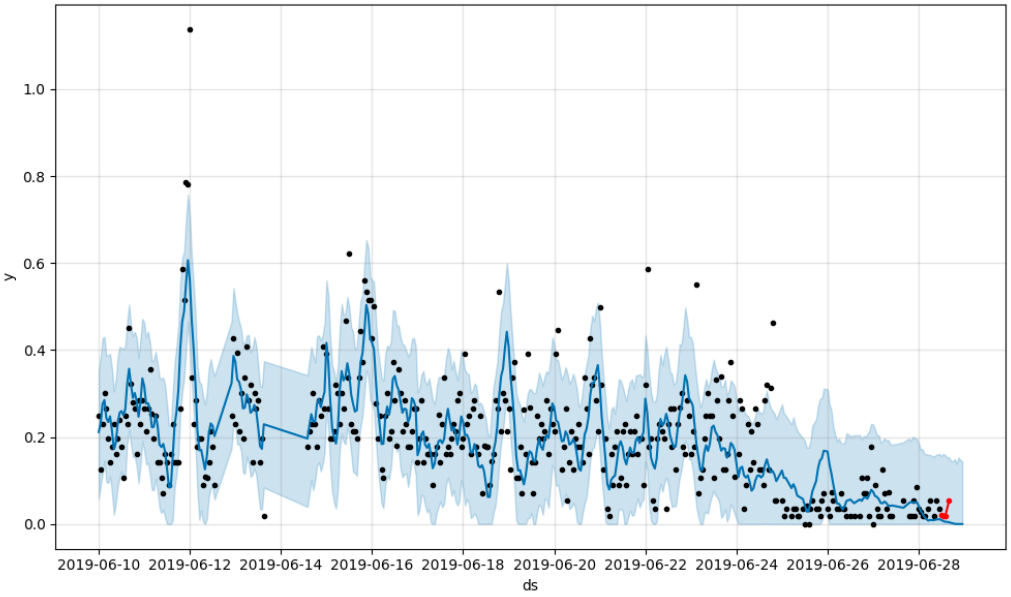
Prima di eseguire il training viene effettuata la validazione del modello, in modo da scegliere i valori da assegnare ai sopra citati iper-parametri: viene effettuata quindi la cosiddetta model selection. La fase di validazione verrà ampiamente discussa nell'apposita sezione; occorre però notare fin da subito, che la fase di model selection viene effettuata una volta a settimana per ogni metrica di ogni host, in modo da ammortizzare il costo dell'operazione considerata.

Dopo aver effettuato il training, utilizzando gli iper-parametri scelti durante la fase di model selection, si passa alla predizione di uno o più punti; le predizioni effettuate dovranno essere confrontate con i valori reali assunti dalla metrica sotto analisi.

Per far ciò, per impostazioni di default, Prophet genera degli intervalli di incertezza: il modello creato, oltre a predire il valore esatto della metrica in un certo istante temporale, fornisce una forchetta di possibili valori (che comprende anche il valore predetto) che possono essere assunti dalla metrica in esame; tale range di valori tiene conto dei cambiamenti di trend rilevati durante la fase di training e di eventuali osservazioni affette da rumore [19]. a tal proposito è stata incrementato il valore del parametro interval_width al 90% (il valore di default è 80%), il quale definisce la quantità di dati che andranno ad influire sulla determinazione degli intervalli di incertezza [20]; ciò consente di ottenere rilevazioni meno sensibili e quindi meno inclini all'individuazione di falsi positivi. Quindi, dato il valore reale di una metrica ad un certo istante temporale t, si controlla se esso cade o meno all'interno del range di valori dell'intervallo di incertezza individuato da Prophet al tempo t; nel caso in cui il valore attuale della metrica stia fuori dall'intervallo di incertezza, allora viene generato un alert di comportamento anomalo.

A discrezione dell'utilizzatore del software realizzato per questo lavoro di tesi, è possibile effettuare un ulteriore controllo prima di sollevare la suddetta anomalia: è possibile accertarsi che, nel periodo in cui si è verificato il comportamento sospetto della metrica, compaiono categorie di protocolli/eventi particolari: contatti con host catalogati come malware o collegati ad attività di mining, protocolli di accesso remoto e protocolli sconosciuti.
Nel caso in cui all'anomalia rilevata da Prophet venga associata la presenza dei protocolli sopra definiti, la possibilità che si tratti di un vero positivo aumentano notevolmente; d'altra parte questa tecnica può portare alla rilevazione di falsi negativi; queste ultime considerazioni verranno analizzate in seguito nel dettaglio.

L'immagine seguente mostra il grafico raffigurante i risultati del fitting su circa tre settimane di dati e dalla predizione di dodici nuovi punti nel futuro, della metrica bytes ricevuti.



comportamento della metrica bytes ricevuti

I punti in nero rappresentano i valori della metrica sui quali il modello di Prophet viene allenato; la curva in blu approssima la serie temporale reale, ed è ottenuta tramite il processo di fitting inizialmente (fino all'istante temporale in cui è presente l'ultimo punto in nero), e successivamente grazie alla fase di predizione (dall'ultimo punto in nero (escluso), in poi) ; l'area in celeste corrisponde al range di valori assunti degli intervalli di incertezza. I punti in rosso sono osservazioni reali della metrica in esame e possono essere confrontati con la curva blu sottostante

(valori predetti): in tal caso i valori osservati non sono anomali, poiché ricadono nel range di valori definiti dagli intervalli di incertezza. Si noti la stagionalità e il trend della metrica analizzata, che vengono appresi in modo abbastanza preciso dal modello di regressione.

In estrema sintesi, l'approccio utilizzato da Prophet nella rilevazione di anomalie di rete presenta i seguenti principali vantaggi:

- Flessibilità: è possibile considerare stagionalità con diverse periodicità, facendo varie assunzioni sul trend della serie;
- Non occorre l'interpolazione di dati eventualmente mancanti;
- Fitting molto efficiente;
- (iper-)Parametri umanamente interpretabili, che consentono di migliorare la capacità di predizione del modello creato, facendo assunzione sulla natura della serie temporale da analizzare.

4.3. Allarmi e mitigazione

Generalmente, a seguito della rilevazione di un'anomalia viene generato un'alert e/o viene intrapresa un'azione correttiva automatica.
Nel primo caso, l'allarme generato deve servire all'amministratore di rete o ad altri sistemi automatici di terze parti per prendere atto di quanto avvenuto, analizzare la situazione ed eventualmente applicare le dovute azioni che consentano di risolvere il problema che si è presentato; ovviamente, il formato dell'allarme deve dare informazioni più dettagliate possibili su ciò che si è verificato.
Se invece si dispone di un sistema di mitigazione, esso risponde in maniera del tutto automatica all'anomalia rilevata, consentendo quindi di risolvere il problema in maniera autonoma, senza il supporto umano.
Spesso i due approcci presentati vengono usati insieme, come viene fatto in questo lavoro di Tesi.
Rilevata un'anomalia, il sistema realizzato lancia un'alert con le seguenti informazioni:

- **Tipo dell'alert**: si può avere sia un allarme che segnala l'inizio dell'anomalia, sia uno che ne sancisce il rientro (fine dell'anomalia rilevata)
- **Tipo dell'anomalia**: fornisce il nome dell'anomalia rilevata;
- **Host/MAC**: indica l'indirizzo IP (Internet Protocol) o l'indirizzo MAC (Media Access Control) su cui è stata rilevata l'anomalia;
- **Id dell'interfaccia**: identificatore dell'interfaccia di rete sulla quale è avvenuta la rilevazione;
- **Data**: mostra la data in cui è stata rilevata l'anomalia, nel formato YYYY-MM-DDThh:mm:ssZ;
- **Metodo della rilevazione**: indica quale tecnica ha evidenziato l'anomalia (Prophet, RSI, ecc.)
- **Valore anomalo rilevato**: mostra il valore anomalo rilevato, quando questo esiste.

Per sintetizzare il tutto, si considerino i seguenti allarmi rilevati durante una sessione di analisi:

TYPE	ANOMALY	HOST/MAC	IF	DATE	METHOD	VAL
START	ping_flooding	[redacted].239@125	0	2019-06-27T20:15:00Z	TRESHOLD	10000.0
START	ping_flooding	[redacted].145@125	0	2019-06-27T10:35:00Z	TRESHOLD	1.1
END	ping_flooding	[redacted].145@125	0	2019-06-27T17:00:00Z	TRESHOLD	
START	ping_flooding	[redacted].180@125	0	2019-06-27T13:40:00Z	TRESHOLD	1.2
END	ping_flooding	[redacted].180@125	0	2019-06-27T13:45:00Z	TRESHOLD	
START	ping_flooding	[redacted].241@125	0	2019-06-27T09:15:00Z	TRESHOLD	1.2
END	ping_flooding	[redacted].241@125	0	2019-06-27T10:00:00Z	TRESHOLD	
START	ping_flooding	[redacted].241@125	0	2019-06-27T10:20:00Z	TRESHOLD	1.8
END	ping_flooding	[redacted].241@125	0	2019-06-27T10:25:00Z	TRESHOLD	
START	ping_flooding	[redacted].241@125	0	2019-06-27T12:35:00Z	TRESHOLD	1.1
END	ping_flooding	[redacted].241@125	0	2019-06-27T12:55:00Z	TRESHOLD	
START	ping_flooding	[redacted].78@125	0	2019-06-27T08:15:00Z	TRESHOLD	1.8
START	ping_flooding	[redacted].234@125	0	2019-06-27T09:25:00Z	TRESHOLD	4.0

Rilevazione di ping flooding

Oltre alla funzionalità di generare allarmi, il software realizzato si propone di mitigare autonomamente alcune anomalie rilevate.

La funzionalità di mitigazione è disabilità per default, ma a discrezione dell'utente può essere attivata. Essa viene applicata ad anomalie che hanno un alto grado di pericolosità, e che vengono rilevate con tecniche che presentano una quantità minima di falsi positivi (valori soglia e Prophet + NDPI). Inoltre, l'attività di mitigazione riguarda gli host locali rilevati anomali per sospetta attività d'attacco: se un host sembra essere sotto attacco DNS flooding, tale problema non verrà mitigato dal sistema; invece se un host è collegato ad attività di DNS exfiltration, il suo IP (IPv4 o IPv6) verrà inserito in una blacklist in modo da non far transitare il

traffico da esso generato. Per far ciò si utilizza la tecnologia eBPF (extended Berkeley Packet Filter), che consente l'iniezione di codice (ovvero bytecode, generato dal compilatore BCC (BPF Compiler Collection)) eseguibile nel kernel di Linux, con particolari restrizioni [21]. In particolare, è stato scritto un programma XDP (eXpress Data Path), che sfrutta la tecnologia eBPF per analizzare in modo veloce i pacchetti che arrivano sull'interfaccia di rete.

4.3.1. XDP

La tecnologia XDP fornisce degli strumenti per analizzare e filtrare i pacchetti di rete direttamente all'interno del kernel. Il codice XDP viene eseguito nella parte bassa dello stack protocollare, consentendo quindi di operare sui pacchetti alla massima velocità possibile [22]. Il codice XDP per essere iniettato all'interno del Kernel viene controllato da un validatore; quindi il codice presenta delle restrizioni: per esempio non sono ammessi cicli ed ogni volta che si accede ai dati raw dei pacchetti è necessario controllare esplicitamente che non si esca dall'area di memoria in cui viene memorizzato il pacchetto stesso [23].

Per questo lavoro di Tesi è stato scritto in XDP un firewall di rete capace di filtrare i pacchetti di rete in arrivo, in base a delle blacklist di indirizzi IPv4, IPv6 e MAC. In particolare, il codice Python eseguito in spazio utente comunica con il codice XDP eseguito in spazio Kernel tramite delle strutture chiave-valore (Tabelle hash); quando viene rilevata un'anomalia che può essere mitigata, si procede a recuperare l'indirizzo dell'host anomalo, il quale viene inserito in una delle mappe sopra citate (a seconda del tipo dell'indirizzo); a questo punto ogni qualvolta arriva un pacchetto sull'interfaccia di rete il processo XDP si preoccupa di controllare se uno degli indirizzi (MAC o IP) del pacchetto si trova in qualche tabella hash; in caso affermativo si procede ad eliminare il pacchetto ricevuto, senza che esso percorra lo stack protocollare. È chiaro che le tabelle hash utilizzate fungono da blacklists per il firewall di rete. Una blacklist per indirizzi IPv4 in XDP è stata così definita:

```
BPF_TABLE("percpu_hash", uint32_t, uint64_t, ipv4Blacklist, 10000);
```

dove:

- **“percpu_hash”** determina il tipo della tabella, ovvero una tabella hash definita per ogni CPU (per migliorare performance);
- **uint32_t** è il tipo della chiave della tabella; siccome è un indirizzo IPv4 esso occupa 32 bit;
- **uint64_t** è il tipo del campo non chiave della tabella; esso è il contatore dei pacchetti filtrati per il relativo campo chiave, che verrà letto periodicamente dallo spazio utente;
- **ipv4Blacklist** è il nome della tabella;
- **10000** è la capienza della tabella hash.

Tramite codice XDP è stato necessario effettuare il parsing dei pacchetti, in modo da poter estrapolare gli indirizzi MAC, IPv4 e IPv6.
Il pacchetto che arriva sull’interfaccia può essere letto tramite la struttura *xdp_md*, che tra i vari campi contiene i puntatori all’inizio e alla fine del pacchetto.

```
struct xdp_md {
        __u32 data;
        __u32 data_end;
        __u32 data_meta;
        /* Below access go through struct xdp_rxq_info */
        __u32 ingress_ifindex; /* rxq->dev->ifindex */
        __u32 rx_queue_index;  /* rxq->queue_index  */
};
```

Per decodificare il pacchetto si possono utilizzare le strutture presenti nel Kernel di Linux; per esempio per parsare l’header Ethernet si può far uso della struttura *struct ethhdr* nel seguente modo:

```
void* data_end = (void*)(long)ctx->data_end;
void* data = (void*)(long)ctx->data;

struct ethhdr *eth = data; //struct header ethernet
uint64_t nh_off = sizeof(*eth);
if (data + nh_off > data_end) return XDP_DROP; //check bounds
uint64_t macIn = mac2u64(eth->h_source);
uint64_t macEg = mac2u64(eth->h_dest);
if(checkMac(&macIn) || checkMac(&macEg)) return XDP_DROP;
uint16_t h_proto = eth->h_proto;
```

In giallo, si può notare il controllo sui limiti del pacchetto sotto analisi. Le funzioni *mac2u64* e *checkMac*, servono rispettivamente a trasformare un MAC codificato come array di caratteri in un intero a 64 bit e a controllare che il MAC del pacchetto non compaia nella corrispettiva blacklist (in tal caso il pacchetto non risale lo stack protocollare, grazie all'azione *XDP_DROP*. Per completezza si riportano le due funzioni sopra descritte;

```
static inline uint64_t mac2u64(unsigned char* mac) {
        uint64_t macVal;
        macVal =(uint64_t)mac[0] << 40  |
                (uint64_t)mac[1] << 32  |
                (uint64_t)mac[2] << 24  |
                (uint64_t)mac[3] << 16  |
                (uint64_t)mac[4] << 8   |
                (uint64_t)mac[5];

        return macVal;
}

static inline int checkMac(uint64_t* mac){
        uint64_t* value = macBlacklist.lookup(mac);
        if(value){
                *value += 1;
                return 1;
        }
        return 0;
}
```

La decodifica degli header IPv4 e IPv6 segue lo schema illustrato per l'header di livello 2.

In estrema sintesi, XDP consente di implementare un semplice firewall di rete capace di processare i pacchetti in arrivo a velocità molto elevate e nel lavoro di Tesi esso è stato usato per bloccare gli indirizzi di host locali considerati malevoli.

4.4. Architettura software

5. Validazione

5.1. Validazione dei modelli

5.2. Validazione performance di rilevazione

6. Conclusioni e lavori futuri

7. Referenze

1. J. Armin, B. Thompson, D. Ariu, G. Giacinto, F. Roli and P. Kijewski, "2020 Cybercrime Economic Costs: No Measure No Solution," 2015 10th International Conference on Availability, Reliability and Security, Toulouse, 2015, pp. 701-710; *http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7299982&isnumber=7299862.*
2. D. A. Effendy, K. Kusrini and S. Sudarmawan, "Classification of intrusion detection system (IDS) based on computer network," 2017 2nd International conferences on Information Technology, Information Systems and Electrical Engineering (ICITISEE), Yogyakarta, 2017, pp. 90-94; *http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8285566&isnumber=8285462.*
3. Kumar, Manish, M. Hanumanthappa, and TV Suresh Kumar. "Intrusion detection system-false positive alert reduction technique." ACEEE Int. J. on Network Security 2.03 (2011); *https://pdfs.semanticscholar.org/b11c/a573f506c66aea0205cedb30162b97a5f74c.pdf.*
4. Martin Roesch. Writing Snort Rules. How to write Snort rules and keep your sanity. Version 1.7; *https://paginas.fe.up.pt/~mgi98020/pgr/writing_snort_rules.htm.*
5. Introduction to Zeek; *https://docs.zeek.org/en/stable/intro/index.html.*
6. P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, E. Vázquez, Anomaly-based network intrusion detection: Techniques, systems and challenges, Computers & Security, Volume 28, Issues 1–2, 2009, Pages 18-28; *https://www.sciencedirect.com/science/article/pii/S0167404808000692.*
7. Mohiuddin Ahmed, Abdun Naser Mahmood, Jiankun Hu, A survey of network anomaly detection techniques, Journal of Network and Computer Applications, Volume 60, 2016, Pages 19-31;

http://www.sciencedirect.com/science/article/pii/S1084804515002891.

8. Mirsky, Yisroel, et al. "Kitsune: an ensemble of autoencoders for online network intrusion detection." arXiv preprint arXiv:1802.09089 (2018). https://arxiv.org/abs/1802.09089.
9. Wikipedia contributors. (2019, May 7). Sensitivity and specificity. In Wikipedia, The Free Encyclopedia. Retrieved 16:25, June 4, 2019; *https://en.wikipedia.org/w/index.php?title=Sensitivity_and_specificity&oldid=895891646.*
10. Mahjabin, T., Xiao, Y., Sun, G., & Jiang, W. (2017). A survey of distributed denial-of-service attack, prevention, and mitigation techniques. International Journal of Distributed Sensor Networks; *https://doi.org/10.1177/1550147717741463*.
11. Nadler, A., Aminov, A., & Shabtai, A. (2019). Detection of malicious and low throughput data exfiltration over the DNS protocol. Computers & Security, 80, 36-53; *https://www.sciencedirect.com/science/article/pii/S0167404818304000.*
12. Relative Strength Index. (7 ottobre 2018). Wikipedia, L'enciclopedia libera; *it.wikipedia.org/w/index.php?title=Relative_Strength_Index&oldid=100177438.*
13. Wikipedia contributors. (2019, July 1). Autoregressive integrated moving average. In Wikipedia, The Free Encyclopedia. *https://en.wikipedia.org/w/index.php?title=Autoregressive_integrated_moving_average&oldid=904398305*
14. Fried R., George A.C. (2011) Exponential and Holt-Winters Smoothing. In: Lovric M. (eds) International Encyclopedia of Statistical Science. Springer, Berlin, Heidelberg. *https://doi.org/10.1007/978-3-642-04898-2_244*
15. Wikipedia contributors. (2019, July 2). Long short-term memory. In Wikipedia, The Free Encyclopedia. *https://en.wikipedia.org/w/index.php?title=Long_short-term_memory&oldid=904463056*

16. Taylor SJ, Letham B. 2017. Forecasting at scale. PeerJ Preprints 5:e3190v2; *https://doi.org/10.7287/peerj.preprints.3190v2*.
17. Wikipedia contributors. (2019, May 31). Limited-memory BFGS. In Wikipedia, The Free Encyclopedia. Retrieved 17:35, June 29, 2019; *https://en.wikipedia.org/w/index.php?title=Limited-memory_BFGS&oldid=899609391*
18. Prophet, Multiplicative Seasonality; *https://facebook.github.io/prophet/docs/multiplicative_seasonality.html*.
19. Prophet, Uncertainty Intervals; *https://facebook.github.io/prophet/docs/uncertainty_intervals.html*.
20. Bartosz Mikulski. Understanding uncertainty intervals generated by Prophet; *https://www.mikulskibartosz.name/understanding-uncertainty-intervals-generated-by-prophet/*.
21. BPF and XDP Reference Guide; *https://cilium.readthedocs.io/en/latest/bpf/*.
22. Introduction to XDP; *https://www.iovisor.org/technology/xdp*
23. XDP programs with eBPF; *https://prototype-kernel.readthedocs.io/en/latest/networking/XDP/end-user/coding.html#special-xdp-ebpf-cases*