

UNIVERSITÀ DEGLI STUDI “G. d’Annunzio”
Chieti-Pescara

SCUOLA SUPERIORE “G. d’Annunzio”

DOTTORATO DI RICERCA IN

Business, Institutions, Markets

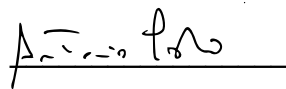
CICLO

XXXVII

**RoSe-T, una verifica automatica delle azioni MANRS per gli
operatori di rete**

**Dipartimento di Scienze Giuridiche e Sociali
Settore Scientifico Disciplinare INFO-01/A**

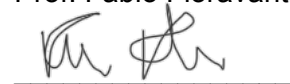
Dottorando
Antonio Prado



Coordinatore
Prof. Marco Angelone



Tutor
Prof. Fabio Fioravanti



Anni accademici 2021/2024

Ringraziamenti

Vorrei ringraziare qui per primo il mio tutor, il professor Fabio Fioravanti, per il tempo e le idee che ha voluto condividere con me, e soprattutto per la guida e la pazienza.

Ringrazio anche i miei compagni di viaggio, fra tutti i ricercatori Mariano Scazzariello e Tommaso Caiazzì con i quali ho condiviso il progetto RoSe-T e che oggi posso chiamare amici.

Una menzione speciale va alla mia famiglia che mi ha sempre sostenuto e incoraggiato nel perseguire le mie passioni.

Dedicato a mamma e babbo

Indice

1 INTRODUZIONE.....	7
2 GLOSSARIO.....	9
3 BGP: ELEMENTI ESSENZIALI E CONTESTO.....	11
3.1 IL SISTEMA AUTONOMO.....	12
3.2 I PUNTI DI INTERSCAMBIO.....	16
3.3 GLI INSTRADAMENTI.....	17
3.4 IL BEST PATH.....	23
3.5 LE POLITICHE DI INSTRADAMENTO.....	25
3.6 LA CONVERGENZA DEL BGP.....	35
3.6.1 Distinzione tra convergenza intra-AS e inter-AS.....	36
3.6.2 Metodi di misura della convergenza.....	36
3.6.3 Impatto delle Politiche di Instradamento.....	37
3.6.4 Interazioni tra politiche locali e convergenza.....	38
3.6.4.1 Route Flap Damping.....	38
3.6.4.2 Politiche incompatibili tra AS adiacenti.....	38
3.6.4.3 Influenza dei filtri sulla propagazione.....	38
3.7 IMPLICAZIONI OPERATIVE.....	39
3.7.1 Impatto della lentezza della convergenza sulla qualità del servizio.....	39
3.7.2 Considerazioni per gli operatori di rete.....	39
3.8 TRADE-OFF TRA STABILITÀ E VELOCITÀ DI CONVERGENZA.....	40
4 LA VERIFICA DEL BGP.....	43
4.1 MACCHINA A STATI FINITI (FSM).....	43
4.1.1 EVENTI ALLA BASE DI FSM.....	43
4.1.1.1 Eventi amministrativi (Administrative events).....	43
4.1.1.2 Eventi a tempo (Timer events).....	44
4.1.1.3 Eventi su connessione TCP (TCP Connection-Based Events).....	44
4.1.1.4 Eventi su messaggio BGP (BGP Message-Based Events).....	44
4.1.2 ILLUSTRAZIONE FSM DEL BGP.....	44
4.1.3 FSM E VERIFICA.....	46
4.1.3.1 Pei - Finite State Machines for BGP [51].....	46
4.1.3.2 Narain - Reconfiguration Planning [52].....	47
4.2 SAT SOLVING E SATISFIABILITY MODULO THEORIES [53, 84, 85, 86, 87]... 48	
4.3 MODELLI DI VERIFICA SYMBOLIC MODEL CHECKING [54].....	49
4.4 MODELLI LOGICI E FORMALI [55].....	50
4.5 FORMALIZZAZIONE DELLE POLITICHE CON METODI LOGICI [55].....	51
4.6 TEORIA DEI GIOCHI E OTTIMIZZAZIONE [56].....	51
5 SICUREZZA DEL BGP.....	53
5.1 ATTACCHI AL BGP.....	53

5.1.1 DIROTTAMENTO DEGLI INSTRADAMENTI.....	53
5.1.2 INSTABILITÀ DEGLI INSTRADAMENTI E DEL BGP.....	54
5.1.3 FUGA DI INSTRADAMENTI.....	55
5.1.4 DISAGGREGAZIONE DEI PREFISSI BGP.....	55
5.1.5 DIROTTAMENTO DELLA SESSIONE BGP.....	56
5.1.6 MANIPOLAZIONE DELLE POLITICHE BGP.....	57
5.1.7 SFIDE PRINCIPALI.....	58
5.2 PROPOSTE E SOLUZIONI PER MIGLIORARE LA SICUREZZA DEL BGP.....	58
5.2.1 AUTENTICAZIONE.....	59
5.2.2 GTSM (THE GENERALIZED TTL SECURITY MECHANISM).....	59
5.2.3 RTBH (REMOTE-TRIGGERED BLACKHOLE).....	59
5.2.4 FLOWSPEC.....	60
5.2.5 SCRUBBING CENTER.....	60
5.2.6 S-BGP (SECURE BGP) [58].....	60
5.2.7 BGPSEC [59].....	61
5.2.8 RPKI (RESOURCE PUBLIC KEY INFRASTRUCTURE) [61].....	61
5.2.9 ARTEMIS [63].....	62
5.2.10 BGP MONITORING E VISUALIZZAZIONE [66].....	62
5.2.11 VERIFICA FORMALE.....	62
5.2.11.1 Piano di controllo e piano dati.....	63
5.2.11.2 Interpretazione Astratta e Scalabilità del Piano di Controllo.....	64
5.2.11.3 Diagnosi e Verifica delle Configurazioni di Rete.....	66
5.2.11.4 Verifica Incrementale e Modifica delle Configurazioni.....	67
5.2.11.5 Verifica Formale e Tecniche di Model Checking.....	69
5.2.11.6 Verifica e Monitoraggio delle Reti Dinamiche.....	70
5.2.12 VERIFICA FORMALE: LIMITI.....	73
5.2.12.1 Complessità computazionale.....	73
5.2.12.2 Scalabilità.....	74
5.2.12.3 Stato transitorio.....	74
5.2.12.4 Mancanza di protezione in tempo reale.....	74
5.2.12.5 Difficoltà logistiche.....	74
5.2.12.6 Dipendenza da modelli e specifiche.....	74
5.2.12.7 Difficoltà con le reti dinamiche.....	74
5.2.12.8 Limitazioni nei tipi di verifica.....	75
5.2.13 CONSIDERAZIONI CONCLUSIVE.....	75
6 MUTUALLY AGREED NORMS FOR ROUTING SECURITY.....	77
6.1 Formalismo di specifica e linguaggio di implementazione.....	77
6.1.1 AZIONE 1.....	79
6.1.1.1 Impedire la propagazione di informazioni errate di instradamento.....	79
6.1.1.2 Relazioni e predicati.....	79
6.1.1.3 Regole e clausole.....	79

6.1.2 AZIONE 2.....	81
6.1.2.1 Bloccare il traffico con indirizzi IP di origine falsificati.....	81
6.1.2.2 Relazioni e predicati.....	81
6.1.2.3 Regole e clausole.....	81
6.1.3 AZIONE 3.....	82
6.1.3.1 Facilitare il coordinamento e la collaborazione con altri operatori.....	82
6.1.3.2 Relazioni e predicati.....	82
6.1.3.3 Regole e clausole.....	83
6.1.4 AZIONE 4.....	83
6.1.4.1 Documentare le proprie informazioni di instradamento.....	83
6.1.4.2 Relazioni e predicati.....	84
6.1.4.3 Regole e clausole.....	84
6.2 RELAZIONI TRA SISTEMI AUTONOMI.....	85
6.2.1 RUOLI BGP.....	86
6.2.2 FORNITORE-CLIENTE.....	87
6.2.3 CLIENTE-FORNITORE.....	91
6.2.4 PEERING.....	92
7 CHIARIMENTI SUL METODO.....	95
7.1 TRANSITI COME CRITERIO DI ESISTENZA DI UN AS.....	95
7.1.1 CUSTOMER-CONE.....	97
7.2 L'ESCLUSIONE DEI PEERING.....	99
7.2.1 AS SPECIALI: I TIER-1.....	101
7.3 CONCLUSIONI.....	105
8 RoSe-T.....	107
8.1 LE CINQUE FASI OPERATIVE.....	109
8.1.1 FASE 1: RACCOLTA E VERIFICA DELLE INFORMAZIONI SUL CANDIDATO.....	109
8.1.2 FASE 2: PARSING DELLA CONFIGURAZIONE.....	110
8.1.3 FASE 3: ANALISI DELLA CONFIGURAZIONE.....	110
8.1.4 FASE 4: EMULAZIONE DELLA TOPOLOGIA MINIMA DELLA RETE... ..	110
8.1.5 FASE 5: VERIFICA DELLA CONFORMITÀ ALLE AZIONI MANRS.....	110
8.2 DETTAGLIO DELLA FASE 1: RACCOLTA E VERIFICA DELLE INFORMAZIONI.....	111
8.3 DETTAGLIO DELLA FASE 5: VERIFICA DELL'AZIONE 1 DI MANRS.....	118
8.3.1 Nota metodologica sulla distinzione tra test e verifica.....	119
8.4 DETTAGLIO DELLA FASE 5: VERIFICA DELL'AZIONE 2 DI MANRS.....	120
8.4.1 Nota metodologica sulla validazione dell'Azione 2.....	121
8.5 ROSE-T: SUPPORTO DEI VENDOR.....	121
9 CONCLUSIONI.....	123
10 APPENDICE A.....	127
10.1 I COMANDI PER LA VERIFICA DELLE AZIONI MANRS 1 E 2.....	127

10.2 I COMANDI PER LA VERIFICA DELL’AZIONE MANRS 4.....	128
11 APPENDICE B.....	131
11.1 DOCUMENTAZIONE DEI PREDICATI USATI NEL CAPITOLO 6.....	131
12 BIBLIOGRAFIA.....	133

1 INTRODUZIONE

Il *Border Gateway Protocol* (BGP) rappresenta uno dei pilastri fondamentali del funzionamento di Internet: consente lo scambio di informazioni di instradamento tra i sistemi autonomi (AS) che compongono la grande Rete. Tuttavia, la natura del protocollo, basata sulla fiducia reciproca tra operatori e su meccanismi complessi di *routing*, espone il BGP a vulnerabilità importanti, come il dirottamento degli instradamenti, la fuga di prefissi e l'instabilità delle sessioni. Questi problemi non solo minacciano l'affidabilità delle comunicazioni, ma possono avere conseguenze economiche e di sicurezza rilevanti.

Questa tesi si inserisce in tale contesto con l'obiettivo di analizzare, verificare e migliorare la sicurezza del BGP attraverso la promozione delle linee guida MANRS (*Mutually Agreed Norms for Routing Security*). In particolare, il lavoro introduce RoSe-T, uno strumento innovativo per la verifica automatica della conformità alle azioni MANRS, pensato per supportare gli operatori di rete nella gestione sicura e resiliente del traffico.

Il percorso di ricerca si sviluppa attraverso una rigorosa analisi del protocollo BGP, dei suoi meccanismi e delle sue vulnerabilità. Successivamente, vengono esplorate soluzioni tecniche, dai modelli di verifica formale alle metodologie basate su *SAT Solving* e *Symbolic Model Checking*, fino alla progettazione e implementazione di RoSe-T. Lo strumento è testato su configurazioni reali, con l'obiettivo di contribuire a ridurre i rischi legati al *routing*.

La tesi si articola come segue: dopo un glossario per chiarire i principali termini tecnici, viene introdotto il protocollo BGP nei suoi aspetti fondamentali. Successivamente, vengono approfondite le problematiche di sicurezza e le soluzioni proposte dalla comunità scientifica e dagli standard internazionali. Nelle sezioni finali si analizzano le funzionalità di RoSe-T, con un approfondimento su metodologia, *test* effettuati e sviluppi futuri. Infine, vengono tratte conclusioni sui risultati ottenuti e sulle possibili evoluzioni del lavoro.

Questo studio intende offrire un contributo significativo al miglioramento della sicurezza del *routing* su Internet attraverso l'adozione di pratiche MANRS facilitata da RoSe-T, uno strumento pratico e innovativo.

2 GLOSSARIO

Come è buona consuetudine al principio di ogni trattazione ecco un breve elenco dei termini e dei concetti che sono impiegati nel testo seguente e che, non sempre, sono d'uso comune.

BGP: il protocollo utile al funzionamento di Internet attraverso lo scambio di informazioni di instradamento.

Sistema autonomo: la cellula fondamentale del tessuto di Internet che, identificata univocamente da un numero, è indispensabile per tracciare i percorsi dei dati sulla Rete.

IP: il protocollo che specifica le modalità con cui i pacchetti di dati vengono indirizzati e trasmessi da un dispositivo sorgente a un dispositivo di destinazione attraverso una o più reti. Oggi sono attive due versioni di IP, la versione 4 (ad esempio 10.0.0.0/8) e la versione 6 (ad esempio 2001:15a8::/32).

TCP/IP: l'insieme di protocolli che specifica come i dati devono essere impacchettati, trasmessi, instradati e ricevuti tra dispositivi in una rete. È progettato per fornire comunicazioni *end-to-end* affidabili e scalabili su reti diverse.

Router: il dispositivo *hardware* o *software* che gestisce il traffico di rete attraverso l'instradamento dei pacchetti sulla base degli indirizzi IP di destinazione. Il *router* decide il percorso migliore per ogni pacchetto e si assicura che i dati raggiungano la loro destinazione in modo efficiente.

CIDR: il sistema che consente di specificare blocchi di indirizzi IP per mezzo di una notazione che combina l'indirizzo di rete con un suffisso che indica la lunghezza del prefisso di rete, per esempio 10.0.0.0/8 oppure 2001:db8::/32.

Best Path: il percorso scelto da un *router* BGP come il più adatto per inoltrare pacchetti di dati a una specifica destinazione IP. Questo percorso viene selezionato utilizzando una serie di criteri o attributi definiti da BGP per garantire che il traffico di rete sia instradato in modo efficiente e affidabile.

RIR: *Regional Internet Registry*, l'entità che supervisiona l'allocazione e la registrazione delle risorse IP pubbliche (blocchi di indirizzi IP e numeri di sistemi autonomi) all'interno di un'area geografica specifica di portata continentale.

LIR/NIR: *Local Internet Registry/National Internet Registry*, l'organizzazione che riceve blocchi di indirizzi IP e numeri AS da un RIR e li distribuisce ai propri clienti o utenti finali. Di solito un LIR è un fornitore di servizi Internet (ISP), un fornitore di servizi di *hosting* o un'azienda grande con necessità di gestire reti IP su larga scala.

3 BGP: ELEMENTI ESSENZIALI E CONTESTO

Il BGP (*Border Gateway Protocol*) nasce nel 1989 come protocollo per lo scambio di informazioni di *routing* tra diversi Sistemi Autonomi (*Autonomous System*, AS). Per questo rientra nella definizione di *Exterior Gateway Protocol* (EGP), a differenza degli *Interior Gateway Protocol* (IGP) che operano all'interno dei sistemi autonomi senza mai travalicarne il perimetro. È un protocollo in produzione, cioè è lo *standard* sul quale si basa e funziona la moderna Internet, per come la conosciamo noi oggi, almeno dal marzo 1995, anno in cui è stata pubblicata la RFC 1771 [\[1\]](#) e [\[2\]](#) dove viene dettagliato il suo funzionamento e dalla quale è possibile estrapolare alcune essenziali peculiarità [\[3\]](#):

- TCP. Le informazioni BGP vengono scambiate attraverso un trasporto affidabile basato su TCP (la porta riservata da IANA allo scopo è la 179).
- PATH VECTOR. Il BGP si basa su un algoritmo di instradamento chiamato “vettore di percorsi” (*path vector*), cioè i messaggi che produce contengono una lista di percorsi dati dai sistemi autonomi che occorre attraversare per raggiungere una certa destinazione (identificata da un prefisso di rete).
- CIDR. Il BGP usa oggi *Classless Inter Domain Routing* (tecnica introdotta nel 1993 da IETF [\[4\]](#)) sia per IPv4, sia (ma non potrebbe essere altrimenti) per IPv6.
- ALGORITMO DI SELEZIONE. Il BGP sceglie il miglior percorso (*best path*) verso una destinazione applicando un rigoroso quanto complesso algoritmo di selezione (a volte arricchito o modificato dalle implementazioni dei *vendor*).
- INGEGNERIA DEL TRAFFICO. Duttile e malleabili criteri di instradamento (*policy routing*) possono condizionare il traffico del BGP sia in entrata, sia in uscita dal sistema autonomo.
- AGGIORNAMENTI IMMEDIATI. I cambiamenti negli instradamenti del BGP vengono notificati se e solo quando avvengono (*triggered updates*) e non su base cronometrica (come ad esempio in un altro protocollo di *routing* dinamico come il RIP).
- FIDUCIA E VOLONTARIETÀ. Il BGP, per come è stato consegnato, confida nel fatto che i sistemi autonomi si interconnettano volontariamente riponendo fiducia l'uno nell'altro (cosa che in seguito ha portato alla creazione di protocolli di sicurezza aggiuntivi). A questo proposito è ormai iconica la descrizione che di Internet fa Randy Bush, studioso della sicurezza degli instradamenti: “*You’re in Hackerville here on the*

Internet. Period,” said Randy Bush, a computer scientist who specializes in routing security. “All of this stuff lacks formal discipline... It’s paint and spackle.” (“Su Internet è come vivere nella città degli hacker. Punto e basta. Tutta questa roba manca della minima regola formale: è vernice e stucco”). [t.d.a.] [5]

3.1 IL SISTEMA AUTONOMO

Si tratta della cellula più piccola che dà vita all’organismo di Internet, paragonabile alla tessera di un mosaico la quale può trovarsi nel centro o ai bordi del quadro ma sempre con almeno un lato, preferibilmente almeno due connesso ad altre tessere (infatti la necessità di avere almeno due rapporti di transito con altri sistemi autonomi è uno dei requisiti che può essere richiesto da un RIR per l’assegnazione di un numero di AS. In questo caso la prassi consolidata dei RIR consiste nel ritenere sufficiente anche la sottoscrizione di un contratto di transito con due distinti sistemi autonomi e non obbligatoriamente pretendere come requisito l’operatività delle interconnessioni) [6].



Fig. 1: I Sistemi Autonomi costituiscono Internet come tessere di un puzzle

Una definizione può essere rintracciata nella RFC 1930, *Guidelines for creation, selection, and registration of an Autonomous System (AS)* [7] dove si legge: “An AS is a connected group of one or more IP prefixes run by one or more network operators which has a SINGLE and CLEARLY DEFINED routing policy (Un sistema autonomo è un gruppo di uno o più prefissi IP gestito da uno o più operatori di rete con una politica di instradamento UNICA e

BEN DEFINITA [t.d.a])”. In questo senso, ciascun sistema autonomo viene visto come una singola entità identificata da un numero che è assegnato agli operatori (LIR o NIR) dai RIR, *Regional Internet Registry*, RIPE (Europa, Asia occidentale ed ex Unione Sovietica), APNIC (Area Asia-Pacifico: Asia Centrale, Sud-Est asiatico, Indocina, Oceania), ARIN (Nord America, Isole Atlantiche), LACNIC (Centro-Sud America, Caraibi), AfriNIC (Africa). Fino al 2007 la rappresentazione di un AS avveniva per mezzo di un numero a 16 bit [8 e 9], ad esempio 64500, ma solo quelli tra 1 e 64511 sono assegnabili pubblicamente (AS 0 è riservato invece per alcuni aspetti di sicurezza del BGP). I valori da 64512 a 65534 [15] non sono assegnabili ad AS pubblici (cioè che si affacciano direttamente su Internet) ma sono riservati per uso privato (ad esempio vengono assegnati dall’ISP ai clienti che usano il BGP come protocollo di accesso alla sua rete). Anche i valori da 64496 a 64511 non sono assegnabili come AS pubblici, ma sono riservati alla documentazione. Indisponibile è pure il numero 112 che è stato destinato allo speciale scopo di ospitare istanze *anycast* di server DNS autoritativi per le risoluzioni inverse degli spazi di indirizzamento IPv4 e IPv6 non instradabili su Internet. Anche il numero di AS 23456 (meglio conosciuto come AS_TRANS) non può essere liberamente assegnato poiché può essere usato per facilitare la comunicazione tra un *router* che non conosce la notazione degli AS a 32 bit e uno che invece fa uso di un AS a 32 bit. L’ultimo valore, il 65535 è riservato [10, 11, 12, 13], dopodiché per mezzo di un numero a 32 bit [14] (ad esempio in formato *asplain* 65551 [10, 15, 8] o, in formato *asdot+*, 1.15), come regolata dalla RFC 5396, *Textual Representation of Autonomous System (AS) Numbers* [16]. Anche nello spazio di 32 bit gli AS hanno delle particolari destinazioni d’uso: infatti l’intervallo 65536 - 65551 è per la documentazione, l’intervallo 4200000000 - 4294967294 (circa 95 milioni di AS) è per uso privato e l’ultimo, il numero 4294967295, è riservato per un eventuale uso futuro.

Più dettagliatamente è necessario considerare un “dentro” e un “fuori” dal punto di vista di un AS e cioè rispettivamente instradamenti *intra-AS* e instradamenti *inter-AS*. Gli instradamenti possono essere classificati anche per la modalità con la quale vengono appresi dai *router*: quando un percorso verso una destinazione viene inserito manualmente, allora si chiamerà “*instradamento statico*” (*static routing*); quando invece i dispositivi apprendono gli instradamenti grazie a un protocollo, allora si parlerà di “*instradamento dinamico*” (*dynamic routing*). All’interno di questa ultima categoria si distinguono: per il cosiddetto instradamento interno al sistema autonomo (*intra-AS*), IGP, *Interior Gateway Protocol* (come RIP [17],

EIGRP [18], OSPF [19 e 20], IS-IS [21]); per l'instradamento esterno tra sistemi autonomi diversi (*inter-AS*), EGP, *Exterior Gateway Protocol* (come appunto BGP).

Quanto al rapporto dei sistemi autonomi tra loro, la questione può diventare piuttosto complessa e coinvolgere dinamiche economiche e politiche, forse al di là dello scopo di questo documento. Tuttavia, volendo rimanere in un contesto tecnico, si può tratteggiare una prima sistemazione basata sul numero di adiacenze di un AS con altri. Quel sistema autonomo che infatti è collegato a un solo altro AS (non importa qui se attraverso una o più connessioni fisiche ridondate) è chiamato STUB e rientra nella categoria dei *single-homed*. Si pensi a un piccolo ISP o a una impresa che non ha necessità di avere una vista completa sugli instradamenti di Internet ma a cui è sufficiente un *gateway of last resort*. Quando invece un AS ha rapporti con almeno altri due sistemi autonomi partecipa alla categoria dei *multi-homed*. Qui è necessario approfondire la tassonomia così che sia possibile classificare questi AS come *transit* o *non transit*, cioè se consentono il transito ad altri AS oppure no.

Un AS dà transito quando instrada pacchetti IP provenienti anche da altri AS e destinati anche ad altri AS. Formalmente definito in [22]: “*Formally, AS u transits traffic for AS v iff $AS\ u$ transits some of its provider or peer routes to AS v , i.e., there is a best route r of u such that r is a provider or peer route of u and $export(v,u)[\{r\}] \neq \{\}$. Now we can determine AS relationships as follows.*

- *ASs u and v have a peering relationship iff neither u transits traffic for v nor v transits traffic for u .*
- *AS u is a provider of AS v iff u transits traffic for v and v does not transit traffic for u .*
- *ASs u and v have a sibling relationship iff both u transits traffic for v and v transits traffic for u ”.*

Proseguendo su questa linea, è possibile individuare anche quei sistemi autonomi che stabiliscono sessioni alla pari con altri AS (*peering*). La più imponente differenza, non a livello tecnico ma su un piano contrattuale, tra un rapporto di *peering* e un rapporto di transito risiede nell'onerosità dell'accordo: solitamente il *peering* viene instaurato a titolo gratuito (*settlement free*), viceversa il transito a titolo oneroso. Tuttavia, nell'industria di Internet gli AS possono assumere molteplici ruoli, cioè uno stesso AS può ricevere transito da due altri operatori e contemporaneamente può configurare centinaia di *peering*, oppure stabilisce solo rapporti di *peering* e nessun transito, oppure ancora riceve tre transiti, ne eroga dieci e instaura una manciata di *peering*. La quantità e la natura delle singole sessioni

riflettono le politiche di instradamento del sistema autonomo e i contratti (livelli minimi di servizio compresi) che gli operatori sottoscrivono in funzione del proprio piano di affari. Infatti, confrontando una sessione BGP paritaria, cioè tra pari (*peer*), e una sessione BGP di transito (*transit*) si rileva facilmente che, da un punto di vista tecnico, sono entrambe soggette alle medesime regole (RFC, sintassi, rappresentazione) ma da un punto di vista “politico” ed “economico” sono assai distanti tra loro. Infatti se nel caso di *peering* entrambi i sistemi autonomi coinvolti convengono di essere più o meno equivalenti in termini di risorse e traffico generato, così non è nel caso di transito dove un AS (più grande) lo vende e un altro (più piccolo) lo acquista. Tutto ciò conduce alla possibilità di classificare gli operatori a seconda delle caratteristiche dell’AS che gestiscono. Quella che oggi è generalmente accettata tra i professionisti del *networking* vede il primo abbozzo nel lavoro *An analysis of internet inter-domain topology and route stability* [23] pubblicato nel 1997. Successivamente, una definizione più strutturata si rinviene nell’articolo di Geoff Huston *Interconnection, Peering, and Settlements* [24] del 1999:

- *Tier-1*: è un AS (esteso a più continenti) in grado di connettersi all’intera Internet senza acquistare servizi di transito da altri operatori (*transit free*). I *Tier-1* hanno tra loro generalmente accordi di *peering* e non utilizzano la *default route* per instradare traffico. L’insieme di questi AS (di numero esiguo: meno di venti nel mondo) è spesso indicato come *Default Free Zone* (DFZ). Il numero dei sistemi autonomi classificabili come *Tier-1* può impercettibilmente variare a seconda della definizione stessa di *Tier-1*, del protocollo preso in esame (IPv4 o IPv6) e dell’algoritmo usato per elaborare le osservazioni di Internet. Ecco un paio di punti di vista: Radar di Qrator Labs, <<https://radar.qrator.net/as-rating#connectivity/1>> e ASRank di CAIDA - Center for Applied Internet Data Analysis, <<https://asrank.caida.org>>.
- *Tier-2*: una rete (per lo più geograficamente localizzata all’interno di un continente o parte di esso) che comunica con altre reti, ma che comunque acquista almeno due transiti IP per poter raggiungere tutta Internet. Possibili esempi sono le reti di grandi ISP nazionali. Anche gli ISP *Tier-2* hanno tra loro generalmente accordi di *peering*. Il *peering* tra ISP *Tier-2* avviene di solito attraverso collegamenti diretti privati (PNI, *Private Network Interconnection*), realizzati utilizzando l’infrastruttura di interconnessione passiva (*Meet-me-room*) messa a disposizione da un punto di interscambio (IXP, *Internet eXchange Point*) [25]. In questo caso si parla di *peering* privato, in contrapposizione al *peering* pubblico dove invece gli ISP si connettono

attraverso il *fabric* dell'IXP. Storicamente conosciuto come NAP, *Network Access Point*, “*Internet eXchange Points (IXPs) are generally considered to be the successors of the four Network Access Points that were mandated as part of the decommissioning of the NSFNET in 1994/95 to facilitate the transition from the NSFNET to the “public Internet” as we know it today.*”

- *Tier-3*: è un AS (di rilevanza regionale o locale) che per raggiungere Internet deve necessariamente acquistare transito da altre reti (almeno due). Di solito i *Tier-3* operano in un ambito geografico ristretto e instaurano degli accordi di *peering* con i *Content Provider*: l'interconnessione avviene spesso all'interno degli IXP e lo scambio dei prefissi attraverso i *Route Server*.

3.2 I PUNTI DI INTERSCAMBIO

Si può definire il punto di interscambio (NAP, *Network Access Point* o *IXP*, *Internet Exchange Point*) come un commutatore di pacchetti in una rete locale (*LAN switch*, detto *IXP Fabric*) al quale si connettono i *router* di alcuni sistemi autonomi interessati a scambiare, principalmente alla pari (*peering*), tra loro gli instradamenti.

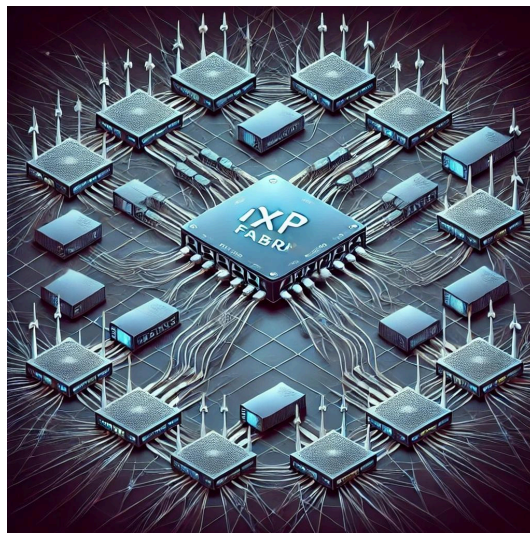


Fig. 2: I punti di interscambio interconnettono router dei Sistemi Autonomi afferenti

Praticamente un utilissimo abilitatore che agevola l'incontro tra AS che potrebbero non conoscersi reciprocamente e che consente, in un unico luogo (fisico o logico), di accedere a una grossa fetta degli instradamenti presenti su Internet. Ma non solo, il NAP interpreta anche un ruolo di facilitatore sul piano tecnico mettendo a disposizione degli afferenti un cosiddetto *route server* [\[26\]](#), cioè una macchina che colleziona, per poi distribuire, gli instradamenti di

tutti i sistemi autonomi che volontariamente aderiscono. Un vero e proprio crocevia di segnalazioni BGP che favorisce tutti i partecipanti evitando loro di configurare interconnessioni multilaterali (*multilateral interconnection*) l'un con l'altro. Se per esempio gli afferenti all'infrastruttura di un NAP fossero cento, in assenza di un *route server* ciascuno di essi dovrebbe configurare sul proprio apparato novantanove sessioni, una con ciascuno degli altri (N-1). Se invece il punto di interscambio mettesse a disposizione un *route server*, allora a ciascun afferente basterebbe configurare una sola sessione con quel dispositivo per vedersi annunciati tutti gli instradamenti dei partecipanti [27].

In Italia i più attivi NAP neutrali sono: MIX, *Milan Internet eXchange*, a Milano; NAMEX, *NAutilus MEditerranean eXchange*, a Roma; TOP-IX, *TOrino Piemonte - Internet eXchange*, a Torino; VSIX, *VeNETo System Internet eXchange*, a Padova. Gli operatori che afferiscono a un NAP possono godere di alcuni vantaggi:

- velocità: la connessione diretta tra due AS, senza passaggi intermedi, minimizza il tempo di latenza dei pacchetti nell'attraversarli migliorando le *performance* di rete specialmente verso tutte le applicazioni interattive o di fruizione di contenuti in tempo reale.
- efficienza: la diversificazione delle connessioni che un operatore ha verso il resto degli afferenti consente di avere un maggior controllo sul *routing* (rafforzando la connettività Internet locale e la sicurezza), di aumentare la ridondanza dell'infrastruttura di rete e quindi il numero di possibili percorsi verso una data destinazione.
- costo: i costi vivi correlati all'afferenza a un NAP (inclusi i costi di interconnessione verso il *data center* dell'IXP e verso il *Fabric*) sono generalmente minori (per unità di banda scambiata) rispetto al costo del transito Internet. Gli accordi di *peering* tra i partecipanti a un punto di interscambio sono nella maggior parte dei casi effettuati a titolo gratuito, cosa che rende l'accesso a Internet più economico e dunque fruibile da un numero più ampio di utenti finali in un determinato Paese o regione (si pensi alle economie in via di sviluppo).

3.3 GLI INSTRADAMENTI

Le informazioni sugli instradamenti possono essere espresse nella forma tipica della modalità CIDR [28 e 29] (tecnica inizialmente conosciuta come “Supernetting” poi anche chiamata

Classless Inter-Domain Routing, cioè IP/lunghezza del prefisso, e vengono chiamati “Prefissi IP” o “Reti”.

Tali reti appartengono a blocchi numerici (IPv4 o IPv6) che IANA distribuisce ai cinque RIR i quali, a loro volta, allocano ai LIR/NIR. Il rispetto della gerarchia nella gestione delle risorse numeriche (gerarchia che da IANA procede ai RIR e da questi ai NIR/LIR) è essenziale per il corretto funzionamento di Internet. È doveroso qui ricordare che infatti il mancato rispetto della gerarchia può provocare, ben che vada, interruzioni di servizio in particolari zone di Internet e, nei casi peggiori, può favorire comportamenti illeciti che possono essere perseguiti come reati.

I LIR/NIR infine operano delle assegnazioni di indirizzi agli utenti. Questi sono i mattoni che costituiscono la *Full Internet Routing Table* (FIRT, chiamata anche GRT, *Global Routing Table*): l'insieme completo degli instradamenti necessari per raggiungere ogni rete IP presente su Internet. Ogni riga di questa tabella rappresenta una rete o un prefisso IP con una destinazione e un percorso specifici, inclusi i *gateway* utilizzabili per raggiungere quella destinazione. L'apprendimento di tale tabella avviene all'interno di una sessione BGP che si instaura tra due *router*. Lo scopo è di garantire la raggiungibilità di qualunque dispositivo provvisto di indirizzo univoco¹.

Ogni sistema autonomo annuncia agli altri AS, con i quali mantiene sessioni BGP, le proprie risorse IP e, eventualmente, quelle dei propri clienti (nel caso in cui fornisca loro un servizio di transito). In questo modo le informazioni sugli instradamenti vengono propagate a tutti gli AS i quali poi useranno un metodo di selezione per eleggere il cosiddetto *best path* verso una determinata destinazione. Lo scambio degli instradamenti avviene attraverso messaggi, propri del BGP, chiamati UPDATE i quali contengono le informazioni di *routing*: nuove reti da aggiungere alla tabella o vecchie reti da eliminare. In questo meccanismo si può apprezzare l'essenza del protocollo BGP e la sua unicità rispetto agli altri protocolli *intra-AS*. Si prenda un BGP *speaking router* e se ne esamini la FIRT: a ciascuna destinazione (rete) del mondo

¹ Una eccezione che qui conferma la regola è la modalità di indirizzamento *anycast*, infatti a proposito dell'univocità degli indirizzi IP, come ben spiegato in [31]: “*IP anycast is an IP addressing mode (v4 or v6) whereby multiple geographically dispersed hosts are assigned the same IP address, with the result that IP routing delivers packets destined to the address to the nearest such host. This works without any changes to unicast routing as routers do not distinguish between multiple routes to multiple different hosts and multiple routes to the same host*”) [30]

Internet viene associato un percorso (o più di uno a seconda delle circostanze), AS_PATH, che indica quanti e quali sistemi autonomi è necessario attraversare per raggiungere quella determinata risorsa. Eccone un esempio:

```
1.1.1.0/24 57866_13335
1.1.1.0/24 24441_13335
1.1.1.0/24 53767_174_174_13335
1.1.1.0/24 3333_13335
1.1.1.0/24 1351_13335
1.1.1.0/24 20912_3257_13335
1.1.1.0/24 8283_13335
1.1.1.0/24 3303_13335
1.1.1.0/24 20130_6939_13335
1.1.1.0/24 3549_3356_13335
1.1.1.0/24 3356_13335
1.1.1.0/24 852_13335
1.1.1.0/24 101_13335
1.1.1.0/24 2497_13335
1.1.1.0/24 4901_6079_13335
1.1.1.0/24 7018_6453_13335
1.1.1.0/24 49788_13335
1.1.1.0/24 1221_13335
1.1.1.0/24 701_13335
1.1.1.0/24 3257_13335
1.1.1.0/24 3561_209_3356_13335
*>1.1.1.0/24 6939_13335
```

Si legge così: secondo il punto di vista del *router* interrogato in un preciso istante nel tempo (nell'esempio riportato si tratta di 128.223.51.103 appartenente al sistema autonomo 3582), ventidue sono i percorsi che conducono alla rete 1.1.1.0/24 residente nel sistema autonomo 13335. Tra essi, l'algoritmo di selezione del *best path* (interno al *router* 128.223.51.103) ha eletto l'ultimo della lista cioè quello che transita per il sistema autonomo 6939. Per dirla diversamente, un *host* che si trovasse nel sistema autonomo 3582 e desiderasse fruire di servizi erogati da un *server* numerato con un IP della rete 1.1.1.0/24 annunciata dall'AS 13335, sarebbe obbligato a inviare e ricevere pacchetti IP attraverso il percorso che transita

dall'AS 6939 il quale mantiene una sessione BGP attiva e funzionante con il *router* 128.223.51.103 e, ovviamente, con il sistema autonomo 13335. Quindi, richiamando la nozione di BGP come protocollo *Path Vector*, è possibile definire l'AS_PATH come un elenco ordinato (*vector*) degli AS, attraversati dall'annuncio di una rete, che costituisce il percorso (*path*). Dunque il BGP consente attraverso specifici strumenti di manipolare il processo di selezione del *best path* [32] così da soddisfare le esigenze dell'operatore di rete.

I *best path* determinati dal processo di selezione BGP non vengono necessariamente installati nella tabella di *routing* IP (RIB, *Routing Information Base*) e quindi trasferiti nella FIB (*Forwarding Information Base*) per essere usati nel *forwarding* del traffico. Infatti, nel caso in cui uno stesso prefisso venga annunciato al *router*, oltre che via BGP, anche in un altro modo (protocollo di *routing* dinamico, *routing* statico, prefisso direttamente connesso), il *router* sceglie quale annuncio installare nella RIB sulla base di un grado di preferenza che il *router* assegna, localmente, a ciascun protocollo di *routing*. La regola usata da tutti i costruttori è quella di preferire l'annuncio del protocollo che ha grado di preferenza più basso.

Per i dispositivi Cisco, nel contesto del processo di selezione del *best path*, si parla di *administrative distance*, per i *router* del *vendor* Juniper ci si riferisce a *route preference value*, per i prodotti Huawei si preferisce utilizzare la locuzione di *routing protocol preference*.

Citato più volte, è necessario soffermarsi sul messaggio *UPDATE* che, come ricordato in precedenza, veicola i contenuti. Si presenta così:

+	-----+
	Withdrawn Routes Length (2 octets)
+	-----+
	Withdrawn Routes (variable)
+	-----+
	Total Path Attribute Length (2 octets)
+	-----+
	Path Attributes (variable)
+	-----+
	Network Layer Reachability Information (variable)
+	-----+

Uno stesso messaggio *UPDATE* può contenere contemporaneamente informazioni sia relative a instradamenti da eliminare (*withdrawn route*) sia a instradamenti da aggiungere (NLRI, *Network Layer Reachability Information*) alla tabella interna al *router*. In più, ciascun campo citato può contenere multipli valori. Si consideri il seguente esempio:

NLRI	Next_Hop	AS_PATH
203.0.113.0/24	198.51.100.1	64496_65551_64511_65536

Il NEXT_HOP rappresenta l'indirizzo IP del *router* più vicino che deve essere attraversato per raggiungere la destinazione desiderata.

Dopo averlo popolato, il messaggio *UPDATE* si presenta con questo contenuto:

+-----+		
		Withdrawn Routes Length
+-----+		
		Withdrawn Routes
+-----+		
		Total Path Attribute Length
+-----+		
	AS_PATH 64496 65551 64511 65536	Path Attributes
	NEXT_HOP 198.51.100.1	
+-----+		
	203.0.113.0/24	NLRI
+-----+		

Altra ipotesi potrebbe essere la seguente:

+-----+		
		Withdrawn Routes Length
+-----+		
	240.0.240.0/24	Withdrawn Routes
+-----+		
		Total Path Attribute Length
+-----+		
		Path Attributes
+-----+		
		NLRI
+-----+		

Oppure una combinazione delle due precedenti:

+-----+		
		Withdrawn Routes Length
+-----+		
	240.0.240.0/24	Withdrawn Routes
+-----+		
		Total Path Attribute Length
+-----+		
	AS_PATH 64496 65551 64511 65536	Path Attributes
	NEXT_HOP 198.51.100.1	
+-----+		
	203.0.113.0/24	NLRI
+-----+		

Una speciale considerazione va rivolta agli attributi del percorso (*path attributes*) i quali si articolano in quattro diverse categorie:

- *WELL-KNOWN MANDATORY*. Attributo imprescindibile che deve essere conosciuto da qualunque BGP *speaking router*;
- *WELL-KNOWN DISCRETIONARY*. Attributo che può essere omesso ma che deve essere conosciuto da qualunque BGP *speaking router*;
- *OPTIONAL TRANSITIVE*. Attributo opzionale che, se presente ma non riconosciuto, deve ugualmente essere trasmesso agli altri BGP *speaking router*;
- *OPTIONAL NON-TRANSITIVE*. Attributo opzionale che, se presente ma non riconosciuto, può essere tranquillamente ignorato e non deve essere trasmesso agli altri BGP *speaking router*.

Di seguito alcuni esempi:

AS_PATH rientra fra gli attributi *well-known mandatory*, come pure *NEXT_HOP* e *ORIGIN* (in tutto sono tre); gli unici due *well-known discretionary* sono *LOCAL_PREF* e *ATOMIC_AGGREGATE*; gli attributi *optional transitive* sono *AGGREGATOR*, *COMMUNITY*, *EXTENDED_COMMUNITY*, *AS4_PATH*, *AS4_AGGREGATOR*, mentre gli *optional non-transitive* sono *MULTI_EXIT_DISC*, *ORIGINATOR_ID*, *CLUSTER_LIST*, *Multiprotocol Reachable NLRI* e *Multiprotocol Unreachable NLRI*. Quindi alla luce di quanto appena documentato ecco il completo schema di messaggio *UPDATE*:

14 byte		Withdrawn Routes Length
240.0.240.0/24		Withdrawn Routes
67 byte		Total Path Attribute Length
ORIGIN	IGP	Path Attributes
AS_PATH	64496 65551 64511 65536	
NEXT_HOP	198.51.100.1	
203.0.113.0/24		NLRI

È opportuno aggiungere che il messaggio *UPDATE* è agnostico rispetto al contenuto del campo NLRI: infatti può contenere indicazioni circa destinazioni sia IPv4 sia IPv6 (non contemporaneamente, è ovvio, poiché cambia il *NEXT_HOP*). E, come corollario, è dunque agnostico anche rispetto al trasporto, cioè è possibile che il campo NLRI contenga

destinazioni IPv6 anche se la connessione TCP viene stabilita tra dirimpettai che usano IPv4 e viceversa.

3.4 IL *BEST PATH*

Come visto nell'esempio precedente, un *router* può ricevere da più parti annunci dello stesso prefisso. Gli annunci di tutti i prefissi scambiati vengono memorizzati in determinate aree di memoria associate alle singole sessioni BGP e quindi sottoposti a eventuali manipolazioni utilizzando gli attributi BGP e/o applicando filtri. Il processo BGP all'interno di un *router* è costituito dai seguenti macro-blocchi:

- *Adj-RIB-in*: sono aree di memoria associate a ciascuna sessione BGP dove vengono memorizzati gli annunci ricevuti (tramite messaggi *UPDATE*) dalle varie sessioni BGP.
- *Input Policy Engine*: è un insieme di politiche di *routing* in ingresso applicate agli annunci ricevuti, costituite da filtri sugli annunci e/o manipolazioni degli attributi BGP.
- *Processo di Selezione BGP*: sceglie, tra gli annunci dello stesso prefisso risultanti a valle delle politiche di *routing* in ingresso, il *best path*, ossia il percorso migliore per raggiungere il prefisso.
- *Loc-RIB*: è una tabella che contiene i *best path* candidati a essere immessi nella tabella di *routing* IP e a essere propagati sulle sessioni BGP.
- *Output Policy Engine*: è un insieme di politiche di *routing* in uscita applicate ai *best path* da propagare sulle altre sessioni BGP, costituite da filtri sugli annunci e/o manipolazioni degli attributi BGP.
- *Adj-RIB-out*: sono aree di memoria associate a ciascuna sessione BGP dove vengono memorizzati gli annunci uscenti da propagare sulle sessioni BGP (messaggi *UPDATE*).

Il processo BGP presente all'interno di ciascun *router* sceglie quindi, tra tutti gli annunci dello stesso prefisso, il percorso migliore (*best path*) e successivamente solo quello viene propagato sulle sessioni BGP, seguendo delle regole ben precise. Il percorso migliore viene selezionato sulla base di una ben determinata sequenza ordinata di scelte modulata su varie metriche o proprietà degli annunci. Questa sequenza ordinata di scelte è denominata processo di selezione e alla fine produce un unico *best path*. L'intuizione suggerirebbe di scegliere il

percorso più breve e poiché il BGP non tiene conto della topologia interna degli AS, si può definire come percorso più breve quello che attraversa il minor numero di AS. Tuttavia altri fattori potrebbero entrare in gioco in questo frangente come ad esempio una maggiore ampiezza di banda disponibile su una particolare sessione, degli accordi commerciali più vantaggiosi con un determinato *transit* AS, ecc.

Tutte le volte che per la stessa destinazione i *router* hanno a disposizione più percorsi diversi, devono far riferimento al seguente algoritmo:

1. Preferire l'instradamento con il valore di *LOCAL_PREF* più alto.
2. Preferire l'instradamento con l'*AS_PATH* più corto.
3. Preferire l'instradamento con il tipo di *ORIGIN* più basso ($\{[0] - \text{IGP}\} < \{[1] - \text{EGP}\} < \{[2] - \text{INCOMPLETE}\}$).
4. Preferire l'instradamento con il valore di *MULTI_EXIT_DISC* più basso.
5. Preferire i percorsi appresi da *external* BGP a quelli appresi da *internal* BGP.
6. Preferire l'instradamento che può essere raggiunto attraverso il percorso più breve verso il *NEXT_HOP*.
7. Preferire l'instradamento appreso dal dispositivo con il *ROUTER_ID* più basso.
8. Preferire l'instradamento appreso dal dispositivo con il *NEIGHBOR_ID* più basso.

Alcune implementazioni presenti sul mercato aggiungono altri criteri selettivi come ad esempio: preferire l'instradamento appreso (e installato nella tabella degli instradamenti) per primo [\[33\]](#), [\[34\]](#), [\[35\]](#).

“*When both paths are external, prefer the path that was received first (the oldest one)*”, criterio implementato da Cisco. E ancora, “*Prefer a primary route over a secondary route. A primary route is one that belongs to the routing table. A secondary route is one that is added to the routing table through an export policy.*”, criterio implementato da Juniper. Infine, “*The route that was learned the earliest is preferred.*”, criterio implementato da Huawei.

Tali criteri mirano a migliorare la stabilità e la robustezza del *routing* dando priorità agli instradamenti primari già validati e integrati nella tabella di instradamento. Si tratta di un metodo utile a evitare oscillazioni nei percorsi. In effetti, senza l'introduzione di tali ulteriori criteri, la robustezza potrebbe non essere garantita: si pensi all'ipotesi di un instradamento che, seppure inizialmente valido, diventi subottimale o addirittura non più valido; oppure si consideri il contesto di una rete che si trovi in uno stato di convergenza ritardata. In questi

casi avere un instradamento valido e installato da più tempo consente di superare senza degrado e tanto più senza disservizio i momenti critici appena accennati.

Ovviamente se il *NEXT_HOP* non è raggiungibile, allora l'instradamento viene ignorato, come pure se vengono implementate delle regole per filtrare via alcuni annunci.

3.5 LE POLITICHE DI INSTRADAMENTO

Una politica di *routing* definisce le regole adottate da un AS per la gestione del traffico entrante (*inbound*) e uscente (*outbound*) e le regole di accettazione e invio degli annunci BGP. Gli strumenti per definire una politica di *routing* sono essenzialmente due: il filtro dei prefissi e la manipolazione degli attributi BGP. Il filtraggio consente a un *router* di scegliere quali prefissi IP accettare e/o propagare. In funzione della direzione di applicazione, il filtraggio può essere di due tipi:

- *Inbound* quando consente di scegliere, tra tutti gli annunci BGP ricevuti sulle varie sessioni BGP, quali accettare e quali rifiutare. Gli annunci accettati partecipano al processo di selezione.
- *Outbound* quando consente di scegliere, tra tutti i *best path* determinati, quali propagare ai *router* con i quali sono attive sessioni BGP.

Il filtraggio è anche utile per scegliere quali prefissi ridistribuire da un protocollo IGP al BGP. Esempi di filtri potrebbero essere: non accettare annunci di prefissi IP con maschera troppo grande (ad esempio superiore a 24 bit); non propagare ad altri AS annunci di prefissi IP ricevuti da un particolare AS (ad esempio per impedire che un AS diventi di transito); scartare tutti gli annunci di prefissi non instradabili su Internet, ad esempio prefissi IP cosiddetti privati o dedicati alla numerazione dei collegamenti locali², eccc (altre ipotesi possono essere le liste nere o le fonti dinamiche destinate a essere per vari motivi scartate). La manipolazione degli attributi BGP consente di variare, secondo le proprie esigenze, il valore degli attributi del BGP. Avere il controllo sui valori degli attributi BGP consente di

² Le reti dello spazio di indirizzamento IPv4 che non possono essere instradate su Internet sono 10.0.0.0/8; 172.16.0.0/12; 192.168.0.0/16 e 169.254.0.0/16. Per lo spazio di indirizzamento IPv6 invece le reti non instradabili su Internet sono fc00::/7; recente aggiunta è rappresentata da 3fff::/20) [11, 36, 37, 38, 39], *Bogon* ("A *Bogon*, plural: *bogons*, is a packet with an IP source address in an address block not yet allocated by IANA or the Regional Internet Registries (ARIN, RIPE, APNIC...) as well as all addresses reserved for private or special use by RFCs) [40, 11, 41]

condizionare le scelte del processo di selezione del *best path* e quindi di definire opportune politiche di gestione del traffico entrante e/o uscente dall'AS.

Innanzitutto si consideri l'attributo *AS_PATH* che contiene la sequenza ordinata dei sistemi autonomi (*AS_SEQUENCE*) dai quali occorre transitare per raggiungere una determinata destinazione. Si tratta di un attributo che finché viene scambiato tra BGP *speaking router* appartenenti allo stesso sistema autonomo (iBGP) non viene modificato; ma quando viene scambiato tra sistemi autonomi diversi (eBGP), allora il *router* che riceve l'annuncio aggiunge (*prepend*) il proprio numero di AS all'inizio (cioè a sinistra) della sequenza. Quest'ultima caratteristica richiede un breve approfondimento. Si può comprendere come la quantità di sistemi autonomi riportata nella *AS_SEQUENCE* sia determinante per il processo decisionale che il BGP usa nel selezionare il miglior percorso verso una destinazione. Di conseguenza si può ipotizzare cosa potrebbe succedere se un AS annunciasse a un *neighbor* una sequenza drogata, per non dire falsa. Normalmente AS 64500 annuncerebbe la rete 241.241.0.0/16 dal *router* R1 al dirimpettaio 192.88.99.1 (AS 64496) sul *router* R2 con questo *AS_PATH*: 64500. Dal punto di vista di AS 64496 quindi la tabella degli instradamenti avrebbe questo aspetto:

Tabella BGP di R2

NLRI	NEXT_HOP	AS_PATH
241.241.0.0/16	192.88.99.33	64500

Si dia il caso che R2 riceva da un terzo sistema autonomo la rete 241.241.0.0/16, allora la tabella potrebbe così trasformarsi:

Tabella BGP di R2 con doppio *path*

NLRI	NEXT_HOP	AS_PATH
241.241.0.0/16	192.88.99.33	64500
241.241.0.0/16	198.51.100.225	64502 64499 64500

Gli utenti del sistema autonomo 64496 che volessero raggiungere una risorsa nel perimetro della rete 241.241.0.0/16, transiterebbero semplicemente per AS 64500. Tuttavia, se quest'ultimo volesse (per qualsiasi motivo) forzare il transito per l'AS 64502, cioè per il percorso che l'algoritmo *path vector* non preferisce, come potrebbe agire? Può senz'altro piegare l'attributo *AS_PATH* a proprio vantaggio usando la tecnica del *prepending* che consiste nell'allungare fittiziamente la *AS_SEQUENCE* aggiungendo, più volte, il proprio numero di sistema autonomo così:

Tabella BGP di R2 con doppio path e prepending

NLRI	NEXT_HOP	AS_PATH
241.241.0.0/16	192.88.99.33	64500 64500 64500 64500
241.241.0.0/16	198.51.100.225	64502 64499 64500

Il risultato è che gli utenti dell'AS 64496 vengono ora forzati a transitare per il percorso annunciato da AS 64502 attraverso il *router* 198.51.100.225 che, agli occhi dell'algoritmo *path vector*, è diventato il più appetibile perché più corto di quello artefatto. Infine è bene precisare che per evitare la creazione di un ciclo continuo (*loop*), quando un *router* riceve un annuncio dove è già presente il proprio numero di sistema autonomo, allora il relativo messaggio di *UPDATE* viene ignorato. Si può passare ora all'attributo *NEXT_HOP* che, in ambito BGP, non è esattamente l'indirizzo IP dell'interfaccia di collegamento del *router* che annuncia l'instradamento. Piuttosto, in eBGP, è l'indirizzo IP del dirimpettaio (*neighbor*), direttamente connesso o no, che annuncia l'instradamento; di conseguenza gli instradamenti che vengono veicolati in iBGP ma appresi da eBGP non vengono modificati e dunque come *NEXT_HOP* recano l'indirizzo IP del *neighbor* che li ha annunciati.

A questo proposito si osservi la ricorsività che questo meccanismo, in piena funzionalità, può generare nel *router* R1:

Tabella BGP di R1

NLRI	NEXT_HOP
203.0.113.0/24	198.51.100.1
240.240.0.0/15	198.51.100.65
250.0.250.0/24	198.51.100.129

Tabella degli instradamenti di R1

NLRI	Gateway
203.0.113.0/24	198.51.100.1
240.240.0.0/15	198.51.100.65
250.0.250.0/24	198.51.100.129
198.51.100.0/26	192.0.2.245
192.0.2.244/30	INT GE0/1
198.51.100.64/26	192.0.2.249
192.0.2.248/30	INT GE0/2
198.51.100.128/26	192.0.2.253
192.0.2.252/30	INT GE0/3

Interfacce di rete su R1

Interfaccia	IP
INT GE0/1	192.0.2.246/30
INT GE0/2	192.0.2.250/30
INT GE0/3	192.0.2.254/30

Per l'instradamento installato nella tabella BGP di R1, 250.0.250.0/24, si apprende il *NEXT_HOP* 198.51.100.129, IP che la macchina dovrebbe essere in grado di raggiungere e di cui ovviamente conosce la posizione. Infatti analizzando la tabella degli instradamenti (*routing table*), si trova che la rete 250.0.250.0/24 è stata installata con l'IP 198.51.100.129 come via d'uscita (*gateway*). È un indirizzo che non appartiene a R1; infatti, scorrendo la *tabella di routing*, si scopre che la rete di appartenenza di quell'IP, 198.51.100.128/26, è raggiungibile attraverso un altro *gateway*, il 192.0.2.253. A sua volta, la rete di appartenenza di quest'ultimo indirizzo, 192.0.2.252/30, è direttamente connessa alla interfaccia *Gigabit Ethernet* numero 3 del modulo numero 0 del *router* R1. Si tratta di una cosiddetta connessione punto-punto (*point to point*) dove, per ogni /30 IPv4, il bit disponibile dispari è assegnato a R1 e il bit disponibile pari al suo *neighbor* BGP. Dall'anno 2000 è anche possibile numerare le connessioni *point to point* usando una rete IPv4 /31, nella quale sono disponibili solo due indirizzi che vengono considerati entrambi assegnabili agli unici due dispositivi presenti nella rete [\[42\]](#). Per concludere si può facilmente dire che R1 può raggiungere uno dei suoi instradamenti, 240.240.0.0/15, inoltrando i pacchetti indirizzati al *NEXT_HOP* 198.51.100.65, attraverso l'interfaccia GE0/2, all'IP 192.0.2.249.

Si può ora spiegare l'attributo *MULTI_EXIT_DISC*, cioè *Multiexit Discriminator*, MED, che diventa particolarmente utile quando un sistema autonomo è collegato a un altro attraverso due sessioni BGP. Nello specifico, può essere usato per influenzare il dirimpettaio a consegnare il traffico laddove si preferisce. Si può chiarire attraverso un esempio che ha per attori protagonisti R1 e R2, legati da un rapporto di *peering* che si esprime con due sessioni

BGP, una tra 192.88.99.1 di R1 e 198.51.100.1 di R2 e l'altra tra 192.88.99.33 di R1 e 198.51.100.65 di R2:

Tabella BGP di R1

NLRI	NEXT_HOP	AS_PATH
203.0.113.0/24	198.51.100.1	64496
240.240.0.0/15	198.51.100.65	64496

Tabella BGP di R2 con MED

NLRI	NEXT_HOP	AS_PATH	MULTI_EXIT_DISC
241.241.0.0/16	192.88.99.1	64500	
242.0.242.0/24	192.88.99.1	64500	
241.241.0.0/16	192.88.99.33	64500	
242.0.242.0/24	192.88.99.33	64500	

Come sarebbe possibile sfruttare MED affinché gli utenti di R2 interessati a risorse erogate da R1 all'interno del prefisso 241.241.0.0/16 passino sempre per 192.88.99.33 anziché per 192.88.99.1? È sufficiente far sì che il messaggio *UPDATE* che R1 trasmette a R2 sia condito dall'attributo MED valorizzato con il numero (*metric*) 100 per il *neighbor* 198.51.100.1 e *metric* 200 per il *neighbor* 198.51.100.65.

Ecco di conseguenza come si modifica la tabella BGP di R2:

Tabella BGP di R2 con MED

NLRI	NEXT_HOP	AS_PATH	MULTI_EXIT_DISC
241.241.0.0/16	192.88.99.1	64500	200
242.0.242.0/24	192.88.99.1	64500	
*>241.241.0.0/16	192.88.99.33	64500	100
242.0.242.0/24	192.88.99.33	64500	

L'algoritmo di selezione del miglior percorso osservato da tutti i BGP *speaking router* rispettosi degli *standard* prevede che venga preferito l'instradamento con il più basso valore MED: si può infatti notare la presenza di un instradamento preceduto dai segni asterisco e maggiore (*>) indicanti il *best path* proprio in corrispondenza della voce con MED 100. È importante precisare comunque che non tutti i sistemi autonomi accettano l'attributo *MULTI_EXIT_DISC*: ricade infatti nella classificazione *optional non-transitive*, cioè attributo opzionale che, se presente ma non riconosciuto, può essere tranquillamente ignorato e non deve essere trasmesso agli altri BGP *speaking router*. In questo contesto, l'instradamento è definito patata bollente (*hot potato*) proprio perché un sistema autonomo tende a liberarsi del traffico (cioè a farlo uscire dalla propria rete) il prima possibile usando il primo punto di uscita che ha a disposizione, senza ragionarci troppo su e dunque senza tenere conto di alcuni eventuali attributi BGP.

Altro attributo indispensabile al controllo degli instradamenti è *LOCAL_PREF*, preferenza che è appunto locale cioè con una valenza solo all'interno di un sistema autonomo; può essere infatti trasmesso solo su sessioni di *internal* BGP (iBGP). Quindi, nel momento in cui si riceve un annuncio da un AS con il quale sia stata stabilita una sessione BGP, si può applicare un valore di preferenza (più alto è il valore, più alta la preferenza) proprio a quello specifico instradamento oppure a tutti gli instradamenti che si apprendono da un determinato eBGP *neighbor*.

Si riprenda l'esempio precedente per esaminare la tabella BGP di R1:

Tabella BGP di R1

NLRI	NEXT_HOP	AS_PATH
203.0.113.0/24	198.51.100.1	64496
240.240.0.0/15	198.51.100.65	64496
240.240.0.0/15	198.51.100.129	64501 64496

È evidente che per la destinazione 240.240.0.0/15 si hanno due diversi percorsi, uno diretto verso l'AS 64496 e l'altro attraverso l'AS 64501. Sembrerebbe più conveniente (nel senso di più breve) la prima occorrenza, tuttavia potrebbe darsi il caso che il collegamento con AS 64501 abbia una capacità di gran lunga maggiore di quella che ha con AS 64496 e che a sua volta AS 64501 abbia un collegamento ad altissima velocità con AS 64496. Tenuto conto di questi ulteriori elementi si può decidere che il percorso più veloce verso 240.240.0.0/15 sia quello con *AS_PATH* più lungo, per questo nel momento in cui si apprende il prefisso da AS 64501, gli si abbina un valore di *LOCAL_PREF* di 150, anziché lasciargli quello predefinito che (seppure non specificato nelle RFC di riferimento) viene tradizionalmente impostato a 100 dalle più diffuse implementazioni attualmente presenti sul mercato [\[43\]](#), [\[44\]](#), [\[45\]](#). Ecco come si trasforma la tabella BGP di R1:

Tabella BGP di R1 con LOCAL_PREF

NLRI	NEXT_HOP	AS_PATH	LOCAL_PREF
203.0.113.0/24	198.51.100.1	64496	100
240.240.0.0/15	198.51.100.65	64496	100
240.240.0.0/15	198.51.100.129	64501 64496	150

Vuol dire che tutto il traffico diretto verso qualsiasi risorsa afferente alla rete 240.240.0.0/15 dovrà transitare per l'AS 64501 nel pieno rispetto di quanto previsto dal processo di instradamento.

Ulteriore utilissimo attributo è *COMMUNITY* che ritrae proprio una comunità: un gruppo che condivide alcune proprietà logiche. Quanto sarebbe comodo raggruppare, nel senso di trattare in modo omogeneo, gli annunci provenienti da clienti? Oppure quelli provenienti da punti di interscambio italiani? Non c'è dubbio che alcuni sistemi autonomi traggano molto giovamento dal corretto uso dell'attributo in questione anche grazie alla sua caratteristica di non essere confinato localmente, infatti appartiene alla specie *optional transitive*, cioè opzionale sì, ma se presente deve essere trasmesso agli altri BGP *speaking router* anche se non riconosciuto. Alcune comunità, date le loro riconosciute ampie utilità e valenza, sono elette come notevoli (*well known*) da alcune RFC e di conseguenza riservate e pubblicate da IANA, *Internet Assigned Numbers Authority*, l'autorità preposta: BGP *Well-known Communities*. Tra queste se ne possono evidenziare almeno un paio: *NO_EXPORT* può contraddistinguere quegli instradamenti che non dovrebbero essere annunciati ad altri sistemi autonomi; *NO_ADVERTISE* è invece più restrittiva poiché potrebbe contraddistinguere quegli instradamenti che non dovrebbero essere annunciati ad alcun *router* dirimpettaio (*neighbor*), cioè neanche a quelli interni all'AS (iBGP). L'attributo *COMMUNITY* è rappresentato da un numero a 32 bit che nella pratica assume l'aspetto di due numeri in base dieci (da 0 a 65535) divisi dal segno di due punti (:). Il primo numero viene solitamente usato per manifestare il numero di sistema autonomo che firma appunto gli instradamenti e il secondo comunica il significato dell'attributo per quello stesso AS.

Tabella BGP di R1 con COMMUNITY

NLRI	NEXT_HOP	AS_PATH	COMMUNITY
203.0.113.0/24	198.51.100.1	64496	64500:39
240.240.0.0/15	198.51.100.65	64496	64500:39
240.240.0.0/15	198.51.100.129	64501 64496	64500:1

In questo esempio, R1, che appartiene all'AS 64500, assegna alcuni attributi *COMMUNITY* agli instradamenti che apprende dalle sessioni eBGP, usando dei numeri che hanno significato solo per chi amministra quel sistema autonomo. Nello specifico il numero 39 contraddistingue instradamenti appresi da un *router* operante su suolo italiano, mentre il numero 1 contraddistingue un instradamento appreso da un *router* operante su suolo statunitense. Nel momento in cui AS 64500 rende conoscibili tali corrispondenze ai suoi clienti, questi ultimi nel ricevere gli instradamenti così classificati hanno la possibilità, a loro volta, di applicare degli ulteriori criteri, come l'accoppiamento a determinati valori di *LOCAL_PREF* o addirittura di assumere la decisione di scartare certi instradamenti.

Le politiche di *routing* possono essere le più varie e trovano la loro naturale applicazione nel caso di AS *multi homed* o di *stub* AS con collegamenti ridondati. Per gli *stub* AS con collegamento singolo ha poco senso parlare di politiche di *routing*, se non per l'aspetto che riguarda il filtraggio degli annunci, poiché gli *stub* AS non hanno vie alternative per la gestione del traffico entrante e/o uscente. Alcuni esempi di politiche di *routing* sono: inviare/ricevere il traffico utilizzando un AS di prima scelta (AS primario) e, in caso di perdita della connettività verso questo, usare un AS di *backup*; bilanciare il traffico entrante e/o uscente tra due o più percorsi; scegliere i percorsi più convenienti in funzione del prefisso. ad esempio, in una configurazione *primario/backup*, potrebbe essere conveniente far passare sul collegamento di *backup* il traffico verso i prefissi locali dell'AS di *backup*. La realizzazione pratica delle politiche di *routing* richiede complesse configurazioni che fanno uso di strumenti specifici messi a disposizione dalle implementazioni del BGP dei vari costruttori. Tra questi rientrano senz'altro i filtri. Infatti le attività per filtrare la tabella BGP, prima che i suoi elementi vengano installati nella tabella degli instradamenti, richiedono la conoscenza di alcuni principi generali e di alcune tecniche particolari.

È bene qui ribadire la distinzione tra due ambiti: traffico in entrata (*inbound*) e traffico in uscita (*outbound*). Si tratta di due diverse prospettive: da una parte (*inbound*) gli annunci che un sistema autonomo comunica ai suoi *neighbor*, dall'altra (*outbound*) gli annunci che riceve dagli altri AS. Se *inbound* è influenzato dagli annunci che facciamo, *outbound* è influenzato dagli annunci che riceviamo. Quindi nel caso in cui nessun utente degli altri sistemi autonomi riesca a raggiungere le risorse di un particolare AS, forse è perché quest'ultimo non sta comunicando a nessuno dove si trovi. Viceversa se alcuni utenti di un AS non riescono a

raggiungere risorse appartenenti ad altri AS, forse è perché quel particolare sistema autonomo sta scartando alcuni annunci in ingresso con la conseguenza che alcune reti non siano presenti nella tabella degli instradamenti. Generalmente, per muoversi in questo contesto è opportuno prima di tutto individuare gli instradamenti oggetto di attenzione, procedere a una attività di accettazione o di scarto di quegli instradamenti e concludere con un'eventuale modifica dei loro attributi BGP.

Come si individuano gli instradamenti interessanti? Gli oggetti utili da osservare sono: l'identificativo di rete relativo a un prefisso IP (203.0.113.0/24), il sistema autonomo al quale il prefisso appartiene (64496), la sequenza di AS (*AS_SEQUENCE*) che occorre attraversare per raggiungere un prefisso, il particolare attributo abbinato a un instradamento (*LOCAL_PREF*), gli instradamenti appresi da un determinato *neighbor* e altri. Questi criteri ovviamente sono valutati tutti, uno dopo l'altro, secondo un meccanismo che mira a filtrare man mano alcuni percorsi o alcuni instradamenti, a seconda della necessità. Le risorse che vengono filtrate possono essere oggetto di scarto vero e proprio oppure di altre attività come per esempio l'abbinamento a un particolare *tag* per un successivo riutilizzo.

3.6 LA CONVERGENZA DEL BGP

Nel contesto del *Border Gateway Protocol* la convergenza può essere definita come il processo tramite il quale la rete raggiunge uno stato stabile dopo una modifica nella topologia, come la perdita di una sessione BGP o un cambiamento negli instradamenti. Questo processo si articola in tre fasi principali:

1. *fault detection* (rilevamento del guasto), cioè il tempo necessario affinché un *router* rilevi l'indisponibilità di un dirimpettaio BGP o l'inaccessibilità di una destinazione. La rilevazione può dipendere dal timeout dei *timer* BGP (es. *Hold Timer*) o da protocolli ausiliari come BFD (*Bidirectional Forwarding Detection*) che offrono tempi di rilevamento inferiori al secondo;
2. propagazione dell'evento, cioè una volta rilevata la modifica, i *router* devono propagare l'informazione a tutti i dirimpettaii BGP. Questo include il ritiro degli instradamenti precedenti e l'annuncio di nuovi percorsi alternativi. La propagazione è soggetta a meccanismi come il *Minimum Route Advertisement Interval* (MRAI), che introduce ritardi intenzionali per limitare l'instabilità degli instradamenti;

3. ricalcolo del percorso (*path recomputation*), cioè ogni *router* coinvolto esegue nuovamente l'algoritmo di selezione BGP per determinare il nuovo *best path*. Questo processo è influenzato da politiche locali, configurazioni del *router* e presenza di più percorsi disponibili.

La convergenza è completata quando tutti i *router* della rete hanno aggiornato correttamente le proprie tabelle di instradamento e non si verificano ulteriori cambiamenti indotti dalla modifica iniziale.

3.6.1 Distinzione tra convergenza intra-AS e inter-AS

La convergenza può essere distinta in due categorie principali a seconda dell'ambito topologico coinvolto:

- convergenza intra-AS che riguarda il ricalcolo e la propagazione degli instradamenti all'interno di un singolo sistema autonomo. In questo caso, il comportamento di BGP è condizionato dalle sessioni iBGP, dall'infrastruttura di trasporto sottostante (IGP) e da ulteriori dispositivi come i *route reflectors*. La convergenza intra-AS tende a essere più veloce e prevedibile, ma è fortemente dipendente dalla topologia iBGP e dalle metriche IGP;
- convergenza inter-AS che coinvolge più sistemi autonomi ed è associata alle sessioni eBGP. Questa convergenza è notoriamente più lenta e complessa, in quanto include meccanismi di *policy-based routing*, interazioni tra operatori con configurazioni e obiettivi eterogenei e un AS-PATH più lungo. Inoltre, il meccanismo del *path hunting*, cioè l'esplorazione sequenziale di rotte subottimali prima di stabilizzarsi su una rotta valida, è più marcato nelle topologie inter-AS.

3.6.2 Metodi di misura della convergenza

La misurazione accurata dei tempi di convergenza è essenziale per valutare la resilienza della rete e confrontare l'efficacia di diversi meccanismi di ottimizzazione. I principali metodi di misura includono:

- RTT (Round Trip Time). Si tratta della misura indiretta della convergenza, utile per monitorare la raggiungibilità di una destinazione attraverso *ping* o *traceroute*. Può

indicare una variazione nel percorso, ma non è sufficiente per isolare il comportamento di BGP.

- Tempo di ritiro e annuncio delle reti. Si calcolano attraverso una analisi temporale dei messaggi *withdraw* e *update* ricevuti dai peer BGP. Il tempo che intercorre tra il primo messaggio di ritiro e il completamento dell'aggiornamento delle tabelle di *routing* fornisce una stima del tempo di convergenza.
- Tracciamenti empirici su *route collector* come quelli offerti da RIPE RIS o *RouteViews*, che permettono di osservare le modifiche nelle tabelle BGP da molteplici punti della rete globale. Queste piattaforme offrono la possibilità di misurare i tempi di propagazione degli instradamenti e di identificare schemi ricorrenti di instabilità.
- Esperimenti attivi eseguiti mediante annunci pilotati di prefissi BGP da AS controllati, per osservare la diffusione e la convergenza degli instradamenti nei sistemi autonomi vicini e remoti.

In tutti i casi, la misurazione della convergenza può variare in base alla posizione topologica dell'osservatore, rendendo necessaria una rete distribuita di punti di misura per ottenere risultati rappresentativi.

3.6.3 Impatto delle Politiche di Instradamento

Le politiche di instradamento in BGP giocano un ruolo centrale nella determinazione del *best path* e, di conseguenza, influenzano direttamente i tempi e la stabilità della convergenza. A differenza di altri protocolli di *routing*, BGP non si basa su metriche numeriche condivise (come costi o latenze), bensì su criteri di selezione locali e spesso arbitrari, stabiliti dagli amministratori di rete. Le politiche locali determinano quali instradamenti accettare, preferire, propagare o filtrare, e sono implementate mediante i meccanismi di seguito riportati.

- *Local preference*: determina la priorità di un instradamento all'interno dell'AS.
- *AS-Path prepending*: aumenta artificialmente la lunghezza dell'AS-PATH per rendere un instradamento meno desiderabile.
- *Prefix-list*, *route-map* e *filter-list*: strumenti per filtrare o manipolare gli annunci in ingresso o uscita.

- *Community e Extended community*: segnali che influenzano la propagazione e il trattamento delle rotte.

Queste politiche, pur essendo fondamentali per garantire sicurezza, bilanciamento del carico e controllo del traffico, possono introdurre effetti collaterali significativi sui tempi di convergenza.

3.6.4 Interazioni tra politiche locali e convergenza

3.6.4.1 Route Flap Damping

Il *Route Flap Damping* (RFD) è una tecnica ideata per mitigare l'instabilità degli instradamenti, penalizzando quelli che oscillano frequentemente. Quando un prefisso viene annunciato e ritirato ripetutamente, BGP gli assegna una penalità crescente; al superamento di una soglia, quella rete viene temporaneamente "silenziosa" e non viene più propagata.

Sebbene efficace nel ridurre il carico sulle tabelle BGP, l'RFD può ritardare significativamente la convergenza, in particolare in scenari dove l'instabilità è transitoria o circoscritta. Per questo motivo, molte reti moderne ne limitano l'uso o adottano configurazioni conservative (es. secondo le raccomandazioni di RIPE-580).

3.6.4.2 Politiche incompatibili tra AS adiacenti

In un ambiente inter-AS, ogni operatore applica politiche indipendenti che possono risultare mutuamente incompatibili. Per esempio:

- l'AS A può filtrare un prefisso ricevuto da AS B, mentre AS C (a valle di A) si aspetta di riceverlo tramite quel percorso;
- due AS possono preferire reciprocamente percorsi che l'altro tende a disincentivare, generando instabilità o *loop* temporanei.

Tali disallineamenti possono causare convergenze cicliche o rallentate, specialmente quando coinvolgono AS *multihomed* o politiche di *backup* con *prepending* aggressivo.

3.6.4.3 Influenza dei filtri sulla propagazione

L'uso di filtri per motivi di sicurezza o gestione delle risorse è comune, ma può limitare la disponibilità di percorsi alternativi durante eventi di ri-convergenza. Di seguito alcuni tra i casi più frequenti.

- *Prefix-list* restrittive: se un AS annuncia un nuovo prefisso per reagire a un guasto, ma i dirimpettai applicano filtri troppo stretti, l’annuncio può essere ignorato o scartato, ritardando il ripristino della raggiungibilità.
- Filtri basati su AS-paths: inibiscono percorsi ritenuti “sospetti” (es. AS-PATH troppo lunghi, o contenenti determinati AS), ma possono anche bloccare instradamenti legittimi che emergono come alternative durante un guasto.
- Filtri basati su *community*: alcune *community* segnalano ai dirimpettai di non annunciare un prefisso a determinati *provider*; in caso di guasto, queste restrizioni possono ostacolare la ri-convergenza globale.

3.7 IMPLICAZIONI OPERATIVE

3.7.1 Impatto della lentezza della convergenza sulla qualità del servizio

La lentezza nella convergenza BGP inter-AS può avere effetti significativi sulla qualità del servizio (QoS) percepita dagli utenti finali e sui livelli di affidabilità offerti dagli operatori di rete. Durante il periodo di convergenza, i *router* possono inviare pacchetti su percorsi obsoleti o non più validi, generando perdita di pacchetti (packet loss); effettuare ricalcoli multipli in caso di *path hunting*, causando ritardi di instradamento (latency spikes); subire brevi episodi di irraggiungibilità verso alcuni prefissi, se i nuovi percorsi alternativi non sono ancora stati propagati o validati.

Questi effetti sono particolarmente critici per servizi in tempo reale come VoIP (*Voice over IP*), videoconferenza, giochi *on-line* o applicazioni in *cloud* che richiedono bassa latenza e alta disponibilità. Anche brevi finestre di instabilità (nell’ordine di 10–30 secondi) possono tradursi in esperienze utente degradate o *timeout* applicativi.

3.7.2 Considerazioni per gli operatori di rete

Per gli operatori di rete, soprattutto ISP di transito o di accesso, la gestione della convergenza inter-AS richiede un equilibrio tra controllo delle politiche e reattività della rete. Alcune buone pratiche operative includono l’ottimizzazione dei *timer* BGP, in particolare l’*Hold Time* e l’*MRAI*, per bilanciare velocità di aggiornamento e stabilità; uso prudente del *prepending*, evitando di penalizzare eccessivamente percorsi di *back-up*; monitoraggio continuo della propagazione dei prefissi, tramite sistemi di *looking glass*, *route collector* e

tracciamenti attivi (es. BGPmon, RIPE RIS); adozione di protocolli ausiliari come BFD, per accelerare il rilevamento dei guasti; test delle politiche prima dell'immissione in produzione, per prevenire interazioni indesiderate tra filtri e instradamento. In ambienti *multihomed*, inoltre, è essenziale progettare scenari di *failover* realistici, testando i tempi di convergenza tra i *provider* e prevedendo *fallback* che non impattino eccessivamente i servizi critici.

3.8 TRADE-OFF TRA STABILITÀ E VELOCITÀ DI CONVERGENZA

Un aspetto fondamentale della progettazione BGP è il *trade-off* tra velocità di convergenza e stabilità del piano di controllo. Accelerare la propagazione degli aggiornamenti può migliorare il *time-to-recovery* in caso di guasto e ridurre il periodo di disservizio percepito dagli utenti. Tuttavia, una convergenza troppo aggressiva può introdurre instabilità e oscillazioni continue (*route flapping*), soprattutto in presenza di guasti intermittenti. Al contrario, misure conservative come l'uso di *Route Flap Damping*, *timer* elevati o filtri rigidi, favoriscono la stabilità ma rallentano la ri-convergenza, peggiorando la resilienza del servizio. Gli operatori devono quindi trovare una configurazione ottimale per il proprio contesto operativo, bilanciando politiche di instradamento personalizzate (esigenze commerciali, sicurezza), reattività della rete in scenari di guasto o riconfigurazione, esigenze dei clienti in termini di SLA (*Service Level Agreement*). Di seguito una tabella di confronto tra metodi orientati alla velocità e alla stabilità nella gestione della convergenza BGP:

Parametro	Metodo orientato alla velocità	Metodo orientato alla stabilità
Rilevamento dei guasti	Protocollo BFD, timer <i>hold</i> ridotti	Solo BGP (<i>timer standard</i>), detection più lenta
Timer MRAI	Valori ridotti o disabilitato	Valori elevati per ridurre la propagazione

<i>Route Flap Damping</i>	Disabilitato o configurato in modalità <i>soft</i>	Abilitato con soglie conservative
<i>Policy di instradamento</i>	Instradamenti alternativi sempre pronti e propagati	Selettività nei percorsi, <i>prepending</i> frequente
Comportamento in caso di <i>failover</i>	Ri-convergenza rapida, anche con percorsi subottimali	Ri-convergenza più lenta, ma stabilità mantenuta
Risorse richieste (CPU/BGP sessioni)	Più aggiornamenti da elaborare, maggior carico	Carico ridotto grazie a filtraggio e <i>damping</i>
Resilienza percepita	Alta: il traffico è reindirizzato rapidamente	Bassa nei primi secondi dopo il guasto
Rischio di instabilità	Alto, se la rete è soggetta a guasti intermittenti	Basso, ma a costo di possibili <i>blackout</i> temporanei
Adatto per	Reti CDN, servizi in tempo reale, ambienti dinamici	<i>Backbone</i> , reti critiche, ambienti conservativi
Controindicazioni	Possibili oscillazioni, aumento dell'instabilità	Tempi lunghi di ripristino della raggiungibilità

4 LA VERIFICA DEL BGP

La verifica del BGP costituisce da anni un campo di studio importante e multidisciplinare (coinvolge infatti strumenti e tecniche provenienti dalla logica, dall'informatica teorica e dai metodi di ottimizzazione) che ha come obiettivo lo studio di metodi per gestire la complessità e risolvere le potenziali vulnerabilità, più volte richiamate in questo testo, negli instradamenti tra sistemi autonomi. In letteratura si riscontrano sistemi e tecniche diversi congegnati per analizzare, verificare e garantire il corretto funzionamento del BGP. Ciascun metodo riporta vantaggi e limiti specifici ma possono offrire soluzioni complementari per verificare la correttezza e la sicurezza del BGP su larga scala. Di seguito una panoramica.

4.1 MACCHINA A STATI FINITI (FSM)

Il BGP si basa su un modello chiaro e semplice e la formalizzazione del suo funzionamento, fin dalla prima stesura della RFC 1105, *A Border Gateway Protocol (BGP)*, giugno 1989 [50], si pone come garanzia di interoperabilità e come *vade mecum* per la risoluzione di eventuali problemi durante l'esercizio. Aggiornamenti al protocollo sono stati apportati dalle RFC 1163 (del 1990) [46], RFC 1267 (del 1991) [47], RFC 1654 (del 1994) [48], RFC 1771 (del 1995) [1], RFC 4271 (del 2006) [2], RFC 6608 (del 2012) [49]. Tale modello viene rappresentato, scientificamente, attraverso un automa a stati finiti, detto anche macchina a stati finiti. In inglese, come riportato anche nei testi delle RFC, si traduce in *Finite State Machine* (FSM). Va comunque precisato che le implementazioni prodotte negli anni dai vari *vendor* presenti possono discostarsi più o meno dalle descrizioni presenti nelle RFC.

4.1.1 EVENTI ALLA BASE DI FSM

I dati in ingresso della FSM sono chiamati EVENTI che si classificano in due macro categorie: obbligatori e facoltativi. Per rendere la trattazione più agevole descriverò qui solo quelli obbligatori.

4.1.1.1 Eventi amministrativi (Administrative events)

ManualStart: l'operatore locale avvia manualmente la connessione con il *neighbor*

ManualStop: l'operatore locale arresta manualmente la connessione con il *neighbor*

4.1.1.2 Eventi a tempo (Timer events)

ConnectRetryTimer_Expires: scadenza dell'attributo di sessione *ConnectRetryTimer* (default a 120 secondi)

HoldTimer_Expires: scadenza dell'attributo di sessione *HoldTimer* (default a 90 secondi)

KeepaliveTimer_Expires: scadenza dell'attributo di sessione *KeepaliveTimer* (con valore definito in *KeepaliveTime*, consigliato di default a $\frac{1}{3}$ di *HoldTimer*)

4.1.1.3 Eventi su connessione TCP (TCP Connection-Based Events)

Tcp_CR_Acked: richiesta del sistema locale di instaurare una sessione TCP con il *neighbor*

TcpConnectionConfirmed: conferma ricevuta dal sistema locale di sessione TCP instaurata con il *neighbor*

TcpConnectionFails: notizia ricevuta dal sistema locale circa l'impossibilità di instaurare una sessione TCP con il *neighbor*

4.1.1.4 Eventi su messaggio BGP (BGP Message-Based Events)

BGPOpen: ricezione di un valido messaggio *OPEN*

BGPHeaderErr: ricezione di un messaggio *BGP* con intestazione non valida

BGPOpenMsgErr: ricezione di un messaggio *OPEN* contenente errori

NotifMsgVerErr: ricezione di un messaggio *NOTIFICATION* contenente un errore di versione

NotifMsg: ricezione di un messaggio *NOTIFICATION* contenente qualunque errore diverso da quello di versione

KeepAliveMsg: ricezione di un messaggio *KEEPALIVE*

UpdateMsg: ricezione di un valido messaggio *UPDATE*

UpdateMsgErr: ricezione di un messaggio *UPDATE* non valido

4.1.2 ILLUSTRAZIONE FSM DEL BGP

Una puntuale descrizione della FSM per il protocollo BGP è al di là dello scopo di questa trattazione, tuttavia è necessario qui fornire uno schema *prêt-à-porter* che, non presente nel testo della RFC, può comunque essere d'aiuto alla comprensione del funzionamento del protocollo. Molto sinteticamente, è così possibile descrivere la FSM: in principio l'automa è in stato di riposo, *IDLE*, e non accetta connessioni BGP, ma può essere svegliato, per così dire, grazie a un evento amministrativo come un *ManualStart*, cioè una concreta azione intrapresa da un amministratore del sistema locale che avvii il processo. L'automa dunque

passa allo stato di connessione, *CONNECT*, durante il quale attende che venga instaurata la connessione TCP. A questo punto le opzioni sono molteplici ma è sufficiente tenere presente che in caso di errore si torna in *IDLE* e che in caso di successo si procede allo stato *OPENSENT*. A questo punto il *neighbor* potrebbe rispondere con un messaggio *OPEN* e allora l'automa invierebbe a sua volta prima un *OPEN* e poi un *KEEPALIVE* per poi passare allo stato *OPENCONFIRM*; oppure la sessione TCP potrebbe fallire e allora si andrebbe nello stato *ACTIVE* che, in fin dei conti, è quello nel quale l'automa tenta di connettersi al *neighbor* attraverso una sessione TCP. Dall'*ACTIVE* si può tornare in *IDLE*, in *CONNECT* o in *OPENSENT* a seconda del verificarsi di altri eventi. Come già accennato, quando è nello stato *OPENSENT*, l'automa attende un messaggio *OPEN* dal *neighbor* e, nel caso vada tutto bene, procede allo stato di *OPENCONFIRM*, invece negli altri casi possibili torna in *IDLE* o in *ACTIVE*. *OPENCONFIRM* è lo stato nel quale si attende un messaggio *KEEPALIVE* o *NOTIFICATION*. Se arriva il *KEEPALIVE* il sistema locale modifica lo stato dell'automa in *ESTABLISHED*, altrimenti torna in *IDLE*. Nello stato *ESTABLISHED* il sistema può scambiare messaggi *UPDATE* e *KEEPALIVE* con il suo *neighbor* pur rimanendo nel medesimo stato. Può scambiare anche messaggi *NOTIFICATION*, ma in questo caso sarebbero forieri di cattive notizie che farebbero tornare l'automa nello stato *IDLE*. Infine è bene tenere sempre presente che ogni BGP *speaking router* deve gestire una macchina a stati finiti per ciascuna sessione BGP configurata e, va da sé, ogni automa segue propri ritmi e propri eventi.

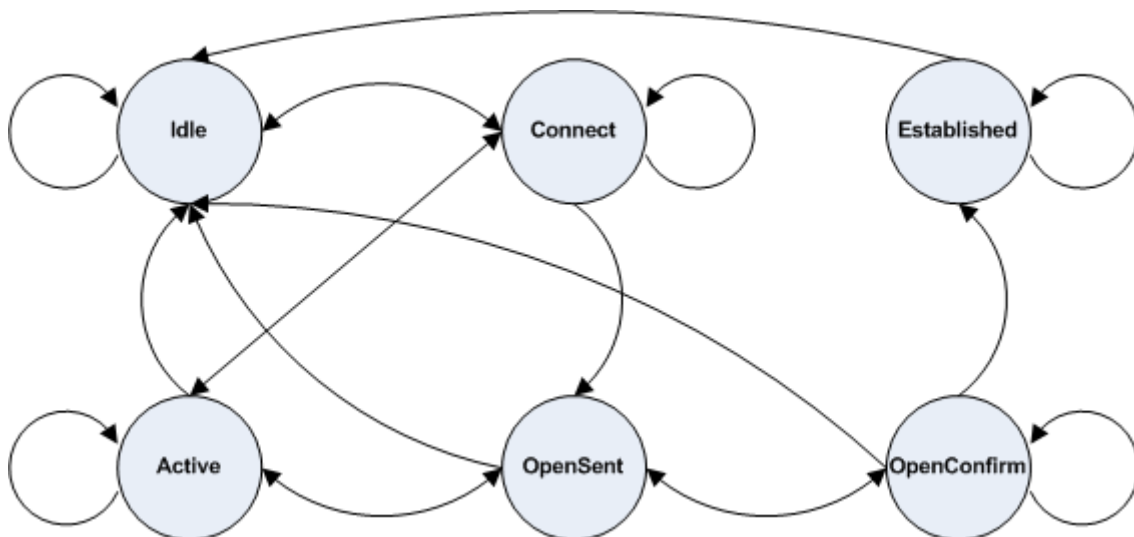


Fig. 3: Rappresentazione dell'automa a stati finiti del BGP

4.1.3 FSM E VERIFICA

Le Macchine a Stati Finiti riescono a rappresentare il BGP come un insieme di stati e transizioni che descrivono il comportamento del protocollo a seguito di eventi esterni. Ciascuno stato corrisponde a una configurazione specifica, mentre le transizioni modellano i cambiamenti sulla scorta di nuovi aggiornamenti o eventi di instradamento. In questo modo è possibile verificare che le transizioni tra stati avvengano correttamente, evitando cicli infiniti o inconsistenze. Alcuni riferimenti fondamentali nella letteratura che trattano l'uso delle FSM nel contesto del BGP includono:

4.1.3.1 Pei - Finite State Machines for BGP [\[51\]](#)

Si analizza il BGP utilizzando FSM per modellare il comportamento delle sessioni BGP e l'elaborazione degli aggiornamenti. Le FSM vengono utilizzate per descrivere i vari stati e le transizioni che un *router* BGP può attraversare in risposta ai messaggi di protocollo. Il lavoro offre una descrizione dettagliata degli stati del protocollo BGP (come Idle, Connect, Active, OpenSent, OpenConfirm, Established) e le transizioni tra questi stati. L'uso delle FSM in questo contesto aiuta a verificare la correttezza della convergenza e a prevenire comportamenti errati durante la fase di instradamento. Le FSM formalizzano le regole del protocollo e garantiscono che solo sequenze legittime di aggiornamenti di instradamento vengano accettate. Messaggi illegittimi, dovuti a guasti o errori di implementazione, possono essere rilevati da violazioni delle transizioni di stato nelle FSM. Un esempio discusso è l'uso di messaggi di withdrawal nel BGP, che notificano ai dirimpettai che un prefisso non è più raggiungibile. Il documento di Pei cita studi che mostrano come molti aggiornamenti BGP agli albori di Internet per come la conosciamo noi oggi fossero "patologici", cioè venivano ritirati percorsi che non erano mai stati annunciati. Questi problemi sarebbero stati rilevati se una FSM di controllo fosse stata implementata. Nel lavoro in questione, ci si riferisce alle FSM che sono state sviluppate per gestire il processo decisionale del BGP in riferimento a un singolo prefisso, cosa che evidentemente semplifica la rappresentazione, concentrandosi sul flusso di messaggi per ciascun prefisso, escludendo l'aggregazione. Nel dettaglio, le FSM sono suddivise in tre tipologie:

- Macchine a stati di input: gestiscono l'elaborazione degli aggiornamenti degli instradamenti provenienti dai dirimpettai BGP.

- Macchine a stati di selezione: gestiscono la selezione del percorso migliore (*best path*) tra le alternative.
- Macchine a stati di output: gestiscono la propagazione degli instradamenti selezionati ai dirimpettai interni (iBGP) ed esterni (eBGP).

Un'importante sezione del documento riguarda le asserzioni derivate dalle transizioni di stato delle FSM. Queste rappresentano regole che devono essere rispettate in ogni esecuzione corretta del BGP. Le principali includono:

- Asserzione 1 (Nuove informazioni): ogni aggiornamento BGP deve contenere nuove informazioni rispetto a quelle già presenti, per evitare un sovraccarico di messaggi inutili.
- Asserzione 2 (Memorizzazione degli aggiornamenti): tutte le informazioni dagli aggiornamenti BGP in ingresso devono essere registrate nella RIB-In (*Routing Information Base*).
- Asserzione 3 (Ritiro di percorsi): un BGP *speaker* non deve ritirare un percorso che non ha mai pubblicizzato o che è già stato ritirato.
- Asserzione 4 (Propagazione iBGP): quando un nuovo percorso viene appreso da un dirimpettaio esterno, questo deve essere propagato a tutti i dirimpettai interni.
- Asserzione 5 (Timer MinRouteAdver): due messaggi UPDATE consecutivi inviati a uno stesso dirimpettaio BGP devono essere separati da un intervallo minimo.
- Asserzione 6 (Modifica più recente): solo l'ultimo percorso selezionato deve essere annunciato al termine dell'intervallo MinRouteAdvertisement.

Le implicazioni di tali asserzioni hanno in particolare un impatto sull'inoltro dei pacchetti, cioè l'assenza di alcune asserzioni potrebbe non influenzare direttamente l'inoltro dei pacchetti, ma potrebbe portare a problemi di sincronizzazione tra dirimpettai BGP; sull'*overhead* dell'instradamento, cioè la mancata osservanza delle asserzioni potrebbe comportare un aumento del traffico BGP, con conseguente sovraccarico di aggiornamenti di instradamenti non necessari.

4.1.3.2 Narain - *Reconfiguration Planning* [\[52\]](#)

Il documento si concentra sul problema della pianificazione della riconfigurazione delle reti, affrontando le complessità nel garantire che i cambiamenti nelle configurazioni di rete non introducano errori o conflitti che possano comprometterne il corretto funzionamento. Quando

si parla di riconfigurazione della rete, ci si riferisce a un processo critico che garantisce la sicurezza e la correttezza della rete durante i cambiamenti di configurazione; evita interruzioni o perdite di servizio dovute a errori nella propagazione di nuove regole; automatizza il processo per ridurre al minimo l'intervento umano e gli errori operativi. Narain formalizza il problema della pianificazione della riconfigurazione utilizzando strumenti di modellazione formale. L'approccio descritto sfrutta modelli matematici per definire esattamente come i cambiamenti nelle configurazioni devono essere applicati e quali vincoli devono essere rispettati. Questo consente la verifica automatica di cambiamenti e aggiornamenti delle regole di rete e la prevenzione di conflitti che potrebbero emergere tra nuove e vecchie configurazioni. Inoltre viene presentata una descrizione dettagliata di come i modelli di rete possono essere rappresentati con strutture come BDD (*Binary Decision Diagrams*) o FSM (*Finite State Machines*) per semplificare la gestione di configurazioni complesse. L'idea principale è di usare queste strutture per verificare la correttezza delle configurazioni proposte prima che vengano implementate. A questo punto il documento presenta alcuni algoritmi utili alla pianificazione della riconfigurazione, come quelli di propagazione delle regole che assicurano che le modifiche vengano applicate in modo sicuro e che non ci siano momenti in cui la rete si trovi in uno stato incoerente; e anche gli algoritmi di verifica che controllano che ogni cambiamento rispetti i vincoli della rete e non causi conflitti tra le politiche esistenti. Altro aspetto interessante trattato nello studio di Narain è l'uso della simulazione per verificare l'impatto di una riconfigurazione prima che venga effettivamente applicata. Questo consente di anticipare eventuali problemi e di risolverli in fase di progettazione.

4.2 SAT SOLVING E SATISFIABILITY MODULO THEORIES [[53](#), [84](#), [85](#), [86](#), [87](#)]

I risolutori SAT sono strumenti che verificano la soddisfacibilità di formule logiche booleane. Nel campo del BGP possono essere utilizzati per affrontare (e risolvere) problemi di instradamento o per verificare che alcune precise condizioni di sicurezza o convergenza siano soddisfatte. Si tratta di un metodo che consente di esplorare spazi di stati molto ampi, cosa che si rivela particolarmente utile per problemi complessi come la risoluzione delle politiche di instradamento (si veda più avanti *Bagpipe* [[99](#)]). I risolutori SMT estendono questa capacità consentendo di includere vincoli più complessi, ad esempio di tipo numerico. Scendendo nel particolare, si può riconoscere che le configurazioni di rete, come quelle

utilizzate per il BGP, possono essere formalizzate in problemi logici che i risolutori SAT possono affrontare. Per esempio, il problema della selezione e propagazione dei percorsi BGP può essere tradotto in formule booleane e risolto come un problema di soddisfacibilità (si veda più avanti *Minesweeper* [85]).

Inoltre, i SAT *solver* possono essere utilizzati per verificare proprietà critiche del BGP, come la convergenza, la stabilità e l'assenza di *loop* nelle tabelle di instradamento. Poiché è noto che il BGP può soffrire di problemi di convergenza dovuti a conflitti tra le politiche di instradamento dei singoli sistemi autonomi coinvolti, un SAT *solver* può aiutare a determinare se una configurazione specifica del protocollo può raggiungere uno stato stabile. E ancora, i SAT *solver* possono essere impiegati per rilevare conflitti di configurazione o anomalie all'interno del BGP, specialmente quando si tratta di risolvere problemi complessi che coinvolgono molteplici sistemi autonomi e le loro interazioni di instradamento. Il rilevamento di percorsi ridondanti, non autorizzati o ambigui può essere formalizzato come problemi SAT e risolto in modo automatico. Lo svantaggio di questi metodi risiede nella complessità computazionale elevata su larga scala, che può limitare l'efficacia per reti di grandi dimensioni o configurazioni complesse.

4.3 MODELLI DI VERIFICA *SYMBOLIC MODEL CHECKING* [54]

Il *model checking* simbolico può essere definito come una tecnica che usa rappresentazioni simboliche per verificare proprietà del sistema. Cioè, invece di enumerare esplicitamente tutti gli stati, SMC sfrutta strutture come OBDD (*Ordered Binary Decision Diagrams*) per rappresentare insiemi di stati. È un metodo in grado di verificare proprietà come la sicurezza e la convergenza del BGP su larga scala. In particolare, il documento di Al-Shaer descrive l'importanza di utilizzare metodi formali per analizzare e verificare le configurazioni di rete, data la complessità e la dimensione delle reti moderne. L'obiettivo principale è garantire che le configurazioni siano corrette e sicure, evitando errori che potrebbero compromettere la sicurezza o la disponibilità della rete. I *Binary Decision Diagrams* (BDD) vengono introdotti come uno strumento efficace per rappresentare le politiche di controllo dell'accesso (ACL) e le configurazioni di sicurezza. Utilizzando BDD, è possibile formalizzare e ottimizzare le configurazioni, oltre a rilevare conflitti o configurazioni errate. Nello specifico, un BDD è una struttura dati che consente di rappresentare funzioni booleane in modo compatto e

permette di effettuare operazioni logiche in modo efficiente. Questo è utile per rappresentare regole complesse di filtraggio del traffico di rete.

Inoltre il documento esplora i vari tipi di conflitti che possono emergere nelle configurazioni di rete, come i conflitti *intra-policy* e *inter-policy* per esempio lo *shadowing* consiste nell'esistenza di una regola che blocca quel traffico che una regola precedente avrebbe permesso; la *redundancy*, cioè una regola ripete le funzioni di un'altra regola, rendendola inutile; l'*exception*, cioè regole che si contraddicono o che portano a comportamenti anomali. In tale contesto, l'uso di BDD consente di rilevare tali conflitti in modo automatico e sistematico. Il documento discute come i metodi formali possono essere usati per verificare le proprietà di sicurezza delle configurazioni di rete. Questo include la verifica della conformità delle configurazioni alle politiche di sicurezza definite e la garanzia che le configurazioni non introducano vulnerabilità o errori. A tale scopo viene presentato lo strumento denominato *ConfigLego* come supporto per l'analisi delle configurazioni di rete tramite BDD. Questo ed altri strumenti possono essere impiegati per automatizzare la verifica e l'ottimizzazione delle configurazioni. La vera sfida in tale contesto è rappresentata dalla scalabilità, cioè le tecniche proposte devono essere in grado di gestire configurazioni di rete di grandi dimensioni in modo efficiente. Utilizzando BDD e altre tecniche di riduzione, il documento mostra come sia possibile ridurre significativamente la complessità dell'analisi senza compromettere la precisione. Tuttavia, nella pratica, può essere complesso configurare modelli di rete realistici per il BGP e interpretare i risultati delle analisi simboliche.

4.4 MODELLI LOGICI E FORMALI [\[55\]](#)

Tali metodi modellano il sistema come un insieme di variabili e vincoli e consentono di esplorare le loro configurazioni possibili per cercare errori logici. *Alloy* è un linguaggio di modellazione leggera che consente di descrivere proprietà complesse di sistemi come il BGP e verificarle automaticamente. È utilizzato per modellare le relazioni tra i vari sistemi autonomi, le politiche di instradamento e verificare la consistenza e la sicurezza delle configurazioni. SPIN usa il linguaggio Promela per verificare modelli concorrenti. Uno degli utilizzi proposti in letteratura è legato alla verifica di problemi di instradamento come lo *Stable Paths Problem*, che è relativo alla convergenza e stabilità del BGP. Questi strumenti consentono di testare configurazioni (astratte) del BGP in modo rapido e scalabile. *Alloy* comunque è limitato nella scala e i modelli devono essere semplificati per problemi molto

grandi. Promela/SPIN invece richiede una modellazione approfondita dei comportamenti di instradamento cosa che può diventare complessa per reti di grandi dimensioni.

4.5 FORMALIZZAZIONE DELLE POLITICHE CON METODI LOGICI [\[55\]](#)

Temporal logic come CTL (*Computation Tree Logic*) e LTL (*Linear Temporal Logic*) vengono impiegati per specificare proprietà temporali e verificarle in sistemi concorrenti. In questo contesto, vengono usate per modellare e verificare che il BGP si comporti correttamente lungo tutte le possibili traiettorie di esecuzione. Tali metodi sono utili per dimostrare che determinate proprietà (come la convergenza o l'assenza di *loop*) siano sempre soddisfatte in ogni possibile esecuzione del protocollo BGP. Prendiamo ad esempio la convergenza [\[82, 83\]](#): dopo un periodo di instabilità (cambiamento della topologia di rete o nuovi instradamenti), il BGP dovrebbe convergere a uno stato stabile in cui non ci sono più aggiornamenti continui. Comunque, modellare sistemi di instradamento complessi con CTL o LTL può essere difficile e la scala del problema potrebbe superare le capacità degli strumenti.

4.6 TEORIA DEI GIOCHI E OTTIMIZZAZIONE [\[56\]](#)

Si tratta di metodi basati sulla teoria dei giochi che modellano le interazioni tra i diversi sistemi autonomi come un gioco strategico, dove ogni AS assume decisioni in grado di influenzare il comportamento globale del BGP. Sono modalità particolarmente utili per analizzare il comportamento cooperativo o competitivo tra AS e per studiare l'impatto delle politiche di instradamento sull'intero sistema. Nel contesto della teoria dei giochi, ogni sistema autonomo può essere visto come un giocatore in un gioco in cui l'obiettivo è massimizzare la propria utilità (ad esempio, minimizzare i costi, massimizzare il *throughput*) tenendo conto delle azioni degli altri AS. Il quadro formale tratteggiato nel lavoro di Voellmy consente di modellare queste interazioni come un gioco, in cui le decisioni dei sistemi autonomi sull'instradamento possono portare a un equilibrio di Nash se le politiche sono progettate correttamente. Inoltre, il bilanciamento del carico è di per sé un problema di ottimizzazione. Il *framework* descritto da Voellmy cerca di trovare una distribuzione ottimale del traffico che minimizzi il rischio di congestione e massimizzi l'efficienza della rete. Vengono applicate tecniche di programmazione lineare e altri metodi di ottimizzazione per

garantire che il traffico sia distribuito su più percorsi in modo da raggiungere questo equilibrio.

In aggiunta, l'ottimizzazione multi-obiettivo viene utilizzata per garantire che le diverse metriche di prestazione, come la latenza e il *throughput*, siano bilanciate in base ai requisiti della rete. Infine, il lavoro discute l'uso di funzioni di costo per modellare il costo dell'instradamento del traffico attraverso diversi fornitori. Queste funzioni di costo possono essere ottimizzate per minimizzare il costo complessivo dell'instradamento, garantendo al contempo il bilanciamento del carico su più percorsi. Il documento esplora anche i compromessi tra obiettivi concorrenti, come la minimizzazione dei costi rispetto alla massimizzazione delle prestazioni, un problema fondamentale nella teoria dell'ottimizzazione.

5 SICUREZZA DEL BGP

Questo è un tema cruciale nella gestione delle reti globali, in particolare perché il BGP costituisce la spina dorsale dell'instradamento tra sistemi autonomi su Internet. La sua progettazione originaria, come già detto, focalizzata più sull'efficienza che sulla sicurezza, lo rende vulnerabile a diversi attacchi e problemi, tra cui il dirottamento di instradamenti, gli attacchi di tipo *man-in-the-middle*, la manipolazione delle informazioni di instradamento. Nel corso degli anni la ricerca accademica e quella industriale hanno proposto numerose soluzioni per migliorare la sicurezza del BGP come ad esempio i filtri per le reti che, sebbene non classificabili propriamente come funzionalità di sicurezza, possono essere comunque pensati come strumenti che aiutano a costruire una architettura sicura. In questo frangente è necessario anche specificare che il principale oggetto delle speculazioni sulla sicurezza è rappresentato solo dalle sessioni BGP di tipo esterno poiché coinvolgono soggetti potenzialmente inaffidabili. Ecco di seguito un panorama delle principali sfide di sicurezza e delle soluzioni proposte nella letteratura.

5.1 ATTACCHI AL BGP

5.1.1 DIROTTAMENTO DEGLI INSTRADAMENTI

Si tratta del BGP *Hijacking* (anche detto *IP Prefix Hijacking*) e rappresenta uno degli attacchi più pericolosi al *Border Gateway Protocol*. Praticamente un sistema autonomo annuncia (intenzionalmente o per imperizia) al proprio dirimpettaio una o più reti che nella realtà appartiene a un altro operatore. La conseguenza è che il traffico destinato al legittimo operatore viene dirottato verso il sistema autonomo sbagliato, con gravi conseguenze per la sicurezza e la disponibilità dei servizi. Si classificano alcuni tipi di dirottamento come quello “completo” dove l'attaccante annuncia lo stesso prefisso IP che appartiene al sistema autonomo legittimo. In quello “incompleto”, invece, l'attaccante annuncia una rete più specifica rispetto al prefisso legittimo. Poiché BGP, in genere, preferisce instradamenti più specifici, il traffico per quel prefisso viene instradato al sistema autonomo dirottatore. Nel dirottamento di tipo “intercettazione”, il traffico viene deviato dal sistema autonomo dirottatore ma poi viene inoltrato alla destinazione legittima dopo essere stato intercettato. Questo consente all'attaccante (praticamente un *Man In The Middle*) di osservare il traffico (come una vera e propria intercettazione), senza interruzioni evidenti del servizio per gli

utenti finali. Infine si può identificare un altro tipo di dirottamento, il cosiddetto “abusivo”, dove il sistema autonomo attaccante annuncia un prefisso IP che non è attivamente usato (annunciato) dal legittimo operatore. È un attacco che non può essere subito rilevato.

Nella recente storia di Internet si sono verificati alcuni importanti episodi di dirottamento come quello del 2017 quando il *provider* russo *Rostelecom* fu accusato di aver dirottato temporaneamente il traffico Internet destinato a diverse importanti reti globali, come *Google* e *Facebook* [80]. O quello occorso nel 2014 quando si è osservato il dirottamento del traffico associato ai minatori di *Bitcoin*, operazione che pare abbia fruttato una grande quantità di moneta virtuale ai dirottatori [81].

5.1.2 INSTABILITÀ DEGLI INSTRADAMENTI E DEL BGP

Il BGP *Route Flapping* è un comportamento in cui un *router* BGP continua ad annunciare e ritirare instradamenti in modo ripetuto e rapido. Questo crea confusione tra i *router* dei dirimpettai e porta a una ripetuta riconfigurazione delle tabelle di instradamento da parte di tutti i sistemi autonomi che sono a conoscenza di quella rete. Il fenomeno può verificarsi per diverse ragioni, come problemi tecnici (instabilità *hardware* o *software* di un *router*); errori di configurazione (configurazioni errate o incoerenti che causano l'annuncio e il ritiro continuo di alcune reti); condizioni di rete intermittenti: (collegamenti instabili o ridondanti che si attivano e disattivano ripetutamente); attacchi intenzionali (un attaccante può sfruttare il *route flapping* come vettore di attacco per sovraccaricare la rete e i sistemi di instradamento). Tale problema di sicurezza può generare gravi ripercussioni sulla stabilità e sulle prestazioni di Internet come per esempio il sovraccarico dei *router* BGP che devono ricalcolare frequentemente le tabelle di instradamento in risposta agli annunci e ritiri continui.

Questo sovraccarica le risorse di calcolo dei dispositivi e può portare al loro congestionamento o blocco; come l'instabilità della rete perché quando il *route flapping* si verifica, le tabelle di instradamento di numerosi sistemi autonomi vengono continuamente aggiornate, causando una perdita di stabilità negli instradamenti; come ancora l'aumento del traffico di controllo, perché si genera un costante flusso di messaggi di aggiornamento BGP UPDATE che devono essere trasmessi e propagati tra tutti i sistemi autonomi dirimpettai; oppure come la latenza e la perdita di pacchetti, cause entrambi di un peggioramento dell'esperienza dell'utente finale; come infine la convergenza lenta perché la rete impiega più tempo a raggiungere uno stato stabile quando i *router* sono costretti a ricalcolare le tabelle di

instradamento. Per alcuni versi questo tipo di attacco può generare un DoS, un vero e proprio “fuori servizio” dei *router*.

5.1.3 FUGA DI INSTRADAMENTI

L’attacco è anche conosciuto come BGP *Route Leakage* e si osserva quando un sistema autonomo propaga instradamenti che non dovrebbe annunciare a determinati dirimpettai, violando di fatto le regole e, probabilmente, anche le politiche di instradamento. Può avvenire a causa di configurazioni errate o intenzionalmente malevole e può causare instradamenti subottimali, congestione e potenziali vulnerabilità alla sicurezza. Premesso che ogni sistema autonomo può avere con altri operatori relazioni di transito (come fornitore o come cliente) o di *peering*, le politiche di instradamento stabiliscono chi può vedere e utilizzare determinati percorsi: infatti, da una parte un sistema autonomo cliente deve annunciare tutte le sue reti al fornitore, comprese quelle di suoi eventuali clienti (ma non quelle apprese dai *peer*), ma contemporaneamente non deve annunciare ai *peer* le reti apprese dal transito; dall’altra parte un sistema autonomo *peer* deve annunciare solo le proprie reti e quelle dei propri clienti, senza propagare cioè instradamenti ricevuti da altri *peer* o da fornitori.

Quando queste regole vengono infrante, accidentalmente o intenzionalmente, allora si verifica la cosiddetta fuga di instradamenti, oltre alla violazione di principi cardine dell’instradamento come il *valley-free routing* (quest’ultimo garantisce che il traffico non venga instradato in modo inefficiente o non redditizio rispettando la seguente regola: un AS non inoltra traffico ricevuto da un *peer* o un *provider* verso un altro *peer* o *provider*, per evitare di agire come transito gratuito; i percorsi scelti sono influenzati dalla struttura economica delle relazioni tra AS, più che dalla distanza geografica o dalla latenza). Le conseguenze sono molteplici, ad esempio il traffico può seguire percorsi non ottimali, aumentando la latenza e il rischio di congestione; oppure un sistema autonomo può venire indotto a fare uso di un percorso più lungo e costoso per instradare il traffico. Questo può esporre i dati a intercettazioni o manipolazioni da parte di malintenzionati.

5.1.4 DISAGGREGAZIONE DEI PREFISSI BGP

Questo attacco si riferisce a una tecnica in cui un sistema autonomo annuncia reti più piccole invece di annunciare un aggregato più grande. Si tratta di un comportamento che, sebbene possa essere una pratica legittima, diventa un attacco quando è sfruttato in modo malevolo o

disattento per ottenere vantaggi indesiderati o causare danni. Si può considerare una disaggregazione legittima quella che occorre a gestire il traffico in modo più preciso, ad esempio per indirizzare determinati instradamenti verso specifici dirimpettai o migliorare il bilanciamento del carico. È senz'altro malevola quella disaggregazione che ha l'effetto di inondare Internet di annunci di prefissi piccoli che altro scopo non hanno se non di sovraccaricare le tabelle di instradamento e di rallentare le operazioni di inoltro del traffico. Si pensi al seguente caso: un operatore ottiene l'assegnazione dal proprio RIR di un prefisso IPv6 /29 e comincia ad annunciarlo su Internet tutto intero. La quantità di righe nella tabella BGP occupata da tale annuncio sarà pari a uno. Ora si ponga il caso che, per necessità di *traffic engineering* o altro tipo di esigenza, lo stesso sistema autonomo cominci ad annunciare la rete /29 e anche un paio di /32. Tutto lecito fin qui, con quantità di righe nella tabella BGP pari a 3. Se tale operatore fosse un malintenzionato, potrebbe sfruttare la regola che consente di annunciare una /48 come rete più piccola nel mondo IPv6, cioè potrebbe annunciare 524.288 reti /48 causando una tanto inutile quanto dannosa esplosione nel numero di righe della tabella BGP. Basti pensare che oggi l'intera tabella BGP IPv6 è popolata da circa 219.000 righe per comprendere la portata dell'attacco che avrebbe come effetto quello di esaurire le risorse *hardware* dei *router* e di provocare interruzioni di servizio.

5.1.5 DIROTTAMENTO DELLA SESSIONE BGP

Un malintenzionato assume il controllo di una sessione BGP esistente tra due *router* BGP, inserendosi nella comunicazione o sostituendosi a uno dei due. Questo attacco può consentire all'attaccante di alterare il flusso del traffico, manipolare le tabelle di instradamento o deviare il traffico per intercettazione o modifiche non autorizzate. Nel dettaglio si osserva che la sessione BGP viene stabilita tra due *router* attraverso il protocollo TCP (di solito su porta 179). Il dirottamento della sessione BGP si basa su una compromissione di questa connessione TCP che può avvenire secondo il seguente possibile copione: l'attaccante intercetta o anticipa la sequenza TCP utilizzata nella sessione BGP tra due *router*; una volta compromessa la sessione TCP, l'attaccante può interrompere la connessione legittima o iniettare falsi aggiornamenti degli instradamenti; può anche prendere il controllo di una delle estremità della sessione e inviare messaggi BGP falsi per aggiornare le tabelle di instradamento del dirimpettaio.

Tale fattispecie risente dunque delle vulnerabilità del protocollo TCP come il *TCP Sequence Number Guessing* dove l'attaccante tenta di indovinare il numero di sequenza TCP usato nella

sessione esistente. Dato che i numeri di sequenza TCP aumentano in modo prevedibile, è possibile indovinare il numero successivo di sequenza e inserire pacchetti nella sessione; oppure come il *TCP Reset (RST) Attack* dove l'attaccante invia un pacchetto TCP con il flag RST (*reset*) impostato allo scopo di terminare la connessione legittima tra i due *router* BGP. Una volta interrotta la sessione, l'attaccante può tentare di stabilire una sessione BGP malevola con uno dei *router* legittimi. Le conseguenze, alcune comuni ad altri tipi di attacco, possono essere: deviazione del traffico (l'attaccante può deviare il traffico attraverso il proprio sistema autonomo per intercettarlo, monitorarlo o modificarlo); *Man-in-the-Middle Attack* (MITM) (l'attaccante può intercettare e inoltrare pacchetti di dati tra due *router* senza che ne siano consapevoli, causa di spionaggio o furto di dati sensibili); modifica delle tabelle di *routing* (l'attaccante può aggiornare le tabelle di instradamento con reti false, compromettendo la stabilità dell'ecosistema e causando potenziali interruzioni); propagazione di aggiornamenti di rete falsi (una volta che la sessione BGP è stata compromessa, l'attaccante può propagare aggiornamenti falsi ai dirimpettai, influenzando reti molto più ampie e causando instabilità su scala globale).

5.1.6 MANIPOLAZIONE DELLE POLITICHE BGP

È un attacco che sfrutta la flessibilità e la complessità delle politiche di instradamento definite dai sistemi autonomi per influenzare le decisioni tra i diversi operatori di Internet. Il BGP è progettato per consentire a ogni sistema autonomo di definire le proprie politiche di instradamento in base alle preferenze locali e alle relazioni commerciali con altri operatori (ad esempio, *peer*, fornitori, clienti). Tuttavia, questa flessibilità può essere sfruttata da malintenzionati per manipolare i percorsi di rete a proprio vantaggio. Praticamente, i sistemi autonomi hanno la capacità di definire politiche per preferire o rifiutare specifici percorsi in base a molteplici criteri come il numero di *hop*, la latenza, i costi, o i rapporti d'affari. Queste politiche sono definite localmente e influenzano la propagazione degli instradamenti nell'intera rete. Un attaccante può manipolare o abusare di queste politiche per influenzare le decisioni di instradamento a livello globale. Altra manipolazione possibile è quella dei prefissi, nel senso che un attaccante può manipolare le politiche per annunciare in modo malevolo percorsi che, pur essendo validi, portano il traffico su *AS_PATH* più lunghi o più costosi, generando inefficienze nella rete. Questo può causare un aumento dei costi per alcuni sistemi autonomi o ridurre le prestazioni complessive.

Ulteriore manipolazione è quella del parametro *Local Preference* che viene impiegato dai *router* per determinare la preferenza di un percorso rispetto ad altri (con validità solo interna allo stesso sistema autonomo). Comunque, un attaccante potrebbe agire sul valore del *Local Preference* per forzare la scelta di percorsi che non sono i migliori, ma che portano il traffico attraverso il proprio AS o altri operatori compiacenti. Altri attributi manipolabili sono *AS_PATH* (un attaccante può accorciarlo artificialmente per far apparire la propria rete come la più vicina alla destinazione, anche se non lo è) e MED, *Multi-Exit Discriminator* (che può essere manipolato per influenzare quale uscita viene scelta quando ci sono più punti di collegamento tra due sistemi autonomi, influenzando la direzione del traffico).

5.1.7 SFIDE PRINCIPALI

Il protocollo BGP, secondo l'idea degli autori, non contempla dei meccanismi per autenticare gli annunci delle reti né per garantire l'integrità delle informazioni che i *router* si scambiano. Questo vuol dire che i sistemi autonomi possono annunciare anche prefissi in modo illegittimo con gravi implicazioni circa la possibilità che il traffico venga dirottato o che i percorsi BGP vengano manipolati. Inoltre, la comunicazione tra *router* che parlano il protocollo BGP avviene in chiaro, cioè senza protezione di tipo crittografico. La possibile conseguenza è che tali scambi siano vulnerabili a intercettazioni (*sniffing*) o modifiche (*tampering*) dei messaggi durante il passaggio. In più, quando un sistema autonomo riceve un annuncio di un instradamento non ha modo di verificare se chi lo annuncia ha realmente l'autorità per farlo oppure se i percorsi annunciati siano stati manipolati. Dunque alla base di una sessione BGP c'è un rapporto di fiducia che, la cronaca lo testimonia, a volte viene tradita.

5.2 PROPOSTE E SOLUZIONI PER MIGLIORARE LA SICUREZZA DEL BGP

La letteratura propone diverse soluzioni per affrontare tali criticità, che spaziano dall'adozione di protocolli crittografici all'introduzione di meccanismi di validazione e monitoraggio.

5.2.1 AUTENTICAZIONE

La RFC 4271 raccomanda l'implementazione della protezione delle sessioni BGP attraverso firme MD5 da verificare agli estremi della connessione TCP. Tale misura non si occupa di crittografare lo scambio di messaggi ma solo di autenticare il mittente dei messaggi attraverso una chiave segreta condivisa da entrambi i dirimpettai di una sessione BGP. Oggi tale previsione, evidentemente vulnerabile, è stata superata e migliorata con la pubblicazione di nuovi meccanismi nella RFC 5925, *The TCP Authentication Option*, giugno 2010. La TCP-AO consente di variare dinamicamente la *password* e di fare uso di più algoritmi per la determinazione della funzione di *hash*, dunque è l'opzione consigliata per tutte le sessioni BGP.

5.2.2 GTSM (*THE GENERALIZED TTL SECURITY MECHANISM*)

Questa misura si basa sul fatto che tipicamente le sessioni BGP vengono stabilite, in TCP, tra *router* vicini, cioè afferenti allo stesso dominio di *broadcast*. Ciò significa che il campo TTL (*Time To Live*) del pacchetto IP subisce un decremento di una sola unità dal suo valore più alto di 255. È il caso di ricordare qui che il TTL definisce il numero massimo di *router* che un pacchetto IP può attraversare prima di venire scartato. Dunque, ad esempio, se il dispositivo A invia al dispositivo B un messaggio via TCP dovendo attraversare quattro *router* nel percorso, all'arrivo il campo TTL sarebbe pari a 251 quindi, dato che è un valore diverso da 0, non viene scartato. Poniamo ora il caso che il dispositivo B abbia impostata una regola tale da scartare tutti i pacchetti IP recanti un valore di TTL minore di 254: ecco come si possa ben comprendere che le connessioni TCP abbiano possibilità di andare a buon fine solo se provenienti da un dispositivo che, al massimo, attraversi un solo *router*. Si tratta di una protezione che è efficace contro attacchi che provengono da remoto e, oltre questo, ha la caratteristica di essere semplice e leggera e perciò facilmente implementabile dagli operatori di rete.

5.2.3 RTBH (*REMOTE-TRIGGERED BLACKHOLE*)

Si tratta di una tecnica che fa uso del BGP contro attacchi di tipo *Denial of Service*. Praticamente, dopo aver individuato l'attaccante (o più d'uno) attraverso un apposito dispositivo (che può essere semplicemente una sonda presente all'interno del sistema autonomo), si invia una segnalazione ai *router* BGP tale che i dati provenienti dall'attaccante (sorgente) e/o diretti verso la vittima (destinazione) devono essere inoltrati al *blackhole*, cioè

scartati. Questo meccanismo è un aiuto che di sicuro si rivela efficace per salvaguardare la rete del fornitore del servizio, meno invece per la rete della vittima che, in effetti, può subire proprio i malfunzionamenti desiderati dall'attaccante.

5.2.4 FLOWSPEC

Questa è una evoluzione della tecnica, forse poco duttile, RTBH appena illustrata. Infatti *Flowspec* <https://www.rfc-editor.org/rfc/rfc8955> ha una maggiore granularità nel trattare il traffico proveniente dall'attaccante e diretto alla vittima nel senso che può agire su microflussi e, volendo, non scartare i pacchetti. I tipi di contenuto che possono essere rilevati sono: prefisso di destinazione, prefisso sorgente, protocollo IP (TCP o UDP), porta (sorgente o destinazione), porta di destinazione, porta sorgente, tipo di ICMP, codice di ICMP, *flag* del TCP, lunghezza del pacchetto, DSCP, frammento. Anche le azioni, come si diceva, possono essere codificate nei seguenti tipi: velocità del traffico in *byte*, velocità del traffico in pacchetti, ridirezione del traffico, modifica della priorità del pacchetto. Dunque entrambi gli elementi flusso/azione possono contribuire a definire una politica di protezione da attacchi DoS tale da non fare il gioco dell'attaccante, ma da consentire al traffico buono di fluire verso la vittima.

5.2.5 SCRUBBING CENTER

Grazie al lavoro di classificazione dei flussi di traffico eseguito da *Flowspec*, è possibile fruire di un ulteriore servizio genericamente chiamato, anche commercialmente, anti-dDoS che consente di raggiungere l'obiettivo di piena *business continuity* nel sistema autonomo della vittima. Questo meccanismo fa uso di un dispositivo conosciuto come *scrubbing center*, cioè capace di pulire, in tempo reale, il traffico cattivo e restituire il traffico buono al destinatario dell'attacco. In tale contesto fa la differenza la capacità del sistema autonomo che eroga l'anti-dDoS di addossarsi e ripulire moli di traffico sempre maggiori. Basti pensare che l'ultimo attacco di questo tipo, mitigato da uno *scrubbing center* [57], ha toccato un picco di 3,8 Tbps e costituisce un *record* a livello mondiale che, presumibilmente, non è destinato a durare molto.

5.2.6 S-BGP (SECURE BGP) [58]

Il Secure BGP è una delle proposte più datate e complete per tentare di risolvere i problemi di sicurezza del BGP. Si basa su certificati PKI (*Public Key Infrastructure*) che servono ad

autenticare i sistemi autonomi e a garantire l'integrità delle informazioni di instradamento. Tra i vantaggi si può dire che S-BGP offra un alto livello di sicurezza, autenticando sia l'origine degli instradamenti sia i percorsi attraversati da questi instradamenti. Tra le limitazioni di questo metodo ci sono la complessità operativa e l'*overhead* computazionale dovuto all'uso della crittografia su larga scala. Infatti la necessità di gestire chiavi crittografiche e verificare certificati può generare un notevole carico computazionale per i *router*, specialmente per reti di grandi dimensioni. Inoltre richiede l'adozione diffusa di una PKI a livello globale, cosa che può essere logisticamente complessa e costosa. Infine, implementare S-BGP ad amplissimo raggio comporta problemi di scalabilità, poiché ciascun sistema autonomo deve gestire chiavi crittografiche e certificati.

5.2.7 BGPSEC [\[59\]](#)

Nata come evoluzione di S-BGP, *BGPsec* è una proposta sviluppata dall'IETF [\[60\]](#) per fornire una sicurezza integrata al BGP attraverso l'autenticazione delle informazioni sugli instradamenti. *BGPsec* si concentra principalmente sull'integrità degli *AS_PATH* cioè dei percorsi. Il lato positivo di *BGPsec* consiste nel fatto di poter garantire che gli AS, quando propagano un percorso, siano effettivamente autorizzati a farlo. La limitazione di tale soluzione, come per S-BGP, risiede nella necessità di un'infrastruttura crittografica, con conseguente aumento dei costi di calcolo e di gestione. In più, un carico crittografico notevole può rallentare l'elaborazione dei messaggi di instradamento nei *router* di bordo. Inoltre, richiede una partecipazione globale per essere efficace, altrimenti i sistemi autonomi che non implementano *BGPsec* potrebbero introdurre vulnerabilità nell'ecosistema di Internet. Infine, anche se protegge i percorsi di instradamento, non offre protezione completa rispetto ad altre vulnerabilità come la validità dei prefissi IP.

5.2.8 RPKI (*RESOURCE PUBLIC KEY INFRASTRUCTURE*) [\[61\]](#)

RPKI è uno standard [\[62\]](#) che consente di verificare se un sistema autonomo abbia o no il diritto di annunciare un determinato prefisso IP. Mediante l'uso di certificati digitali si autentica l'associazione tra un prefisso IP e uno specifico numero di sistema autonomo. In questo senso si deve riconoscere che RPKI è meno complesso da implementare rispetto a *BGPsec* e fornisce al contempo un accettabile livello di sicurezza contro il dirottamento degli instradamenti. Punto debole di questa soluzione è che RPKI si concentra principalmente sull'autenticità dei prefissi IP, ma non affronta la sicurezza completa degli instradamenti.

Questo significa che gli attacchi all'*AS_PATH* restano una vulnerabilità. Inoltre, come per tutte le soluzioni basate su PKI, l'affidabilità dell'intero sistema dipende dal corretto funzionamento degli enti che rilasciano i certificati, cosa che evidentemente introduce potenziali rischi di fallimenti centralizzati. E ancora, un attaccante potrebbe compromettere un'autorità di certificazione e revocare certificati, bloccando così gli instradamenti legittimi.

5.2.9 ARTEMIS [\[63\]](#)

Automatic and real-time detection and mitigation of BGP hijacking, ARTEMIS [\[63, 64, 65\]](#), è una soluzione che si fonda sul monitoraggio attivo e in tempo reale così da consentire ai sistemi autonomi di rilevare e mitigare rapidamente gli attacchi di dirottamento. In questo senso fornisce un metodo proattivo e rapido, riducendo di fatto i tempi di reazione rispetto a soluzioni tradizionali. Tuttavia, sebbene efficace nel rilevamento, ARTEMIS richiede una costante sorveglianza (anche di personale specializzato) e non previene completamente gli attacchi. Infatti è uno strumento reattivo, non proattivo. Rileva e mitiga gli attacchi solo dopo che si sono verificati, ma non impedisce che avvengano. Infine, la qualità della protezione fornita da ARTEMIS è direttamente proporzionale alla qualità dell'infrastruttura di monitoraggio distribuita.

5.2.10 BGP *MONITORING* E VISUALIZZAZIONE [\[66\]](#)

Gli strumenti di visualizzazione e monitoraggio consentono di analizzare in tempo reale il comportamento degli instradamenti BGP e conseguentemente di rilevare anomalie o potenziali attacchi. Queste rilevazioni vengono condotte globalmente con il vantaggio di poter spesso identificare eventuali incongruenze capaci di far scaturire un attacco. Al contempo tale caratteristica rappresenta anche una limitazione poiché il monitoraggio in tempo reale richiede infrastrutture dedicate e non può sempre prevenire o mitigare gli attacchi. Seppure siano utili, e spesso anzi indispensabili per la diagnosi, tali strumenti non sempre si dimostrano sufficienti a mitigare attacchi o errori senza l'impiego di contromisure automatiche.

5.2.11 VERIFICA FORMALE

Utilizzando modelli formali come FSM o strumenti di *model checking*, è possibile garantire che un'implementazione del BGP rispetti rigorosamente le specifiche del protocollo e le politiche di sicurezza. Infatti tali tecniche consentono di identificare vulnerabilità e

comportamenti anomali che potrebbero non essere evidenti durante il normale funzionamento. Possono essere usate anche per simulare scenari complessi di instradamento e verificarne la correttezza in condizioni estreme. Prima della carrellata di alcuni tra i lavori più significativi in questo ambito è utile un richiamo sul piano di controllo e il piano dati all'interno dei *router* e come si relazionano l'un con l'altro.

5.2.11.1 Piano di controllo e piano dati

Nell'ambito dei dispositivi di instradamento BGP, i concetti di *control plane* e *data plane* si riferiscono a due funzioni distinte all'interno del *router*. Il piano di controllo è responsabile della gestione e del controllo del flusso di informazioni all'interno del *router* stesso e tra i *router* della rete. È la parte del *router* che si occupa della creazione e della manutenzione delle tabelle di *routing* e delle politiche di instradamento. Tra le funzioni principali:

- esecuzione dei protocolli di *routing* (come BGP, OSPF, RIP);
- costruzione della tabella di instradamento (*Routing Information Base*, RIB);
- determinazione del percorso migliore per l'invio dei pacchetti basandosi sulle politiche di instradamento;
- comunicazione con altri *router* per scambiare informazioni di instradamento.

Il piano di controllo dunque elabora le informazioni sui percorsi disponibili e determina le decisioni di *routing* da implementare nel piano dati che può essere definito come la parte del *router* responsabile dell'effettivo inoltro dei pacchetti. È la sezione che gestisce il flusso di dati reali attraverso il *router* verso la destinazione finale. Le funzioni principali sono:

- consultazione della *Forwarding Information Base* (FIB), che è derivata dalla RIB del *control plane*;
- inoltro dei pacchetti da un'interfaccia all'altra nel *router*, seguendo le decisioni prese dal *control plane*;
- applicazione di politiche di filtraggio o *Quality of Service* (QoS) se richiesto.

Il piano dati quindi utilizza le informazioni pre-elaborate dal *control plane* per gestire i pacchetti con efficienza e velocità, mantenendo alta la capacità di *throughput*.

Messi a confronto si può dire che il *control plane* prende le decisioni sui percorsi e aggiorna le tabelle di *routing* mentre il *data plane* esegue l'inoltro dei pacchetti basandosi sulle

decisioni prese dal *control plane*. Inoltre il *control plane* gestisce processi più complessi e fa maggiormente uso di CPU, poiché è responsabile dell'elaborazione dei protocolli di *routing*. Il *data plane* invece è progettato per essere efficiente nell'inoltro rapido dei pacchetti, spesso basato su *hardware* dedicato come ASICs (*Application-Specific Integrated Circuits*). Tra loro i due piani interagiscono continuamente poiché da una parte il *control plane* aggiorna periodicamente il *data plane* con nuove informazioni di *routing*, dall'altra il *data plane* opera in modo autonomo per l'inoltro dei pacchetti una volta che è stato configurato dal *control plane*.

Ecco di seguito gli studi raggruppati per aree omogenee.

5.2.11.2 Interpretazione Astratta e Scalabilità del Piano di Controllo

Lo studio di Ryan Beckett, Aarti Gupta, Ratul Mahajan e David Walker dal titolo “*Abstract Interpretation of Distributed Network Control Planes*”, pubblicato nel 2020 [92], tratta specificamente dell'interpretazione astratta per affrontare le sfide di scalabilità nell'analisi del piano di controllo di rete. Nel dettaglio, premesso che i piani di controllo della rete utilizzano protocolli distribuiti per determinare i percorsi di inoltro del traffico e che gli errori possono manifestarsi solo dopo cambiamenti come guasti ai collegamenti, aggiornamenti dalle reti vicine o modifiche alla configurazione, il documento dimostra come l'interpretazione astratta fornisca un metodo per ridurre sistematicamente la complessità della simulazione del piano di controllo. Allo scopo, gli autori hanno sviluppato uno strumento chiamato *ShapeShifter* che viene utilizzato per l'analisi della raggiungibilità, cioè per determinare se le destinazioni della rete possono essere raggiunte con varie configurazioni. Il programma consente di ottenere miglioramenti significativi nei tempi di esecuzione e nell'utilizzo della memoria rispetto ai simulatori più avanzati (come *Batfish*) [93], pur mantenendo un elevato livello di accuratezza: per il 95% delle reti studiate, *ShapeShifter* è stato in grado di prevedere la raggiungibilità con elevata precisione. Inoltre, la tecnica dell'interpretazione astratta consente di sviluppare nuovi tipi di analisi, come l'individuazione di BGP hijacking che, lo ricordiamo, si verifica se un percorso esterno (proveniente da un dirimpettaio) può sovrascrivere un percorso interno, causando un errato instradamento del traffico. Per il 5% restante delle reti esaminate, le complessità delle politiche di instradamento o la granularità persa nell'astrazione sono state le principali ragioni delle imprecisioni.

Ancora, nel contesto della verifica del piano di controllo del BGP è necessario annoverare il documento intitolato “*Fast BGP Simulation of Large Datacenters*” [94] che si concentra sulle sfide della simulazione in reti di grandi *data center* che fanno uso del BGP per gestire l'instradamento dei dati attraverso numerosi *router*. Il protocollo consente a questi dispositivi di scambiare informazioni sui percorsi di rete e di garantire una comunicazione efficiente. Tuttavia, la complessità e le dimensioni di queste reti rendono la simulazione e la verifica un compito impegnativo soprattutto quando si tratta di considerare migliaia di *router*: si tratta di un compito per strumenti in grado di scalare in modo efficace, di non esaurire la memoria o di non andare in *timeout* durante le simulazioni. A questo scopo gli autori introducono lo strumento *FastPlane* progettato per simulare in modo efficiente il BGP: l'innovazione principale consiste nello sfruttare la natura monotona degli annunci di percorso nelle reti di *data center*, caratteristica che consente di utilizzare una versione modificata dell'algoritmo di Dijkstra [95] per simulare la propagazione dei percorsi BGP. La "natura monotona" si riferisce al fatto che, nei *data center* moderni, le politiche di instradamento BGP sono configurate in modo tale che le preferenze degli annunci di percorso non aumentino mai man mano che vengono propagati attraverso la rete. In questo modo lo strumento riesce a ridurre significativamente il tempo di calcolo necessario per verificare il piano di controllo. Il documento dimostra che *FastPlane* è molto più veloce dello strumento concorrente *Batfish* che invece richiede più iterazioni per calcolare il piano di controllo. Gli autori lo hanno testato sui *data center* di un importante *cloud provider* dove ha rilevato diversi errori, tra i quali problemi di *firmware*, errori nei criteri di instradamento e problemi nell'architettura di rete che hanno portato a un bilanciamento del carico e a una tolleranza ai guasti non ottimali.

Il documento intitolato “*Plankton: Scalable Network Configuration Verification through Model Checking*” [96] illustra un nuovo metodo per la verifica delle configurazioni di rete con lo scopo di superare gli ostacoli di scalabilità e complessità degli strumenti esistenti come *Batfish* [93] e *Minesweeper* [85]. Il partizionamento simbolico è la chiave per gestire ampi spazi di istanziazione e il *model checking* a stati espliciti per esplorare in modo efficiente i comportamenti dei protocolli. Plankton combina il calcolo dell'equivalenza dei pacchetti (PEC, raggruppamento dei pacchetti con comportamenti simili) con il controllo del modello a stati espliciti, migliorando la scalabilità e le prestazioni. Tra le ottimizzazioni introdotte ci sono l'hashing degli stati, la riduzione basata sull'ordinamento parziale e il *pruning* basato sulle politiche, che aiutano a ridurre le dimensioni dello spazio degli stati da esplorare durante la verifica. Plankton si affida al *model checker* SPIN [97] per esplorare in modo

esaustivo tutti i possibili stati della rete e identificare eventuali violazioni delle politiche di correttezza. Supporta i principali protocolli di instradamento dinamico come OSPF, BGP e l'instradamento statico, modellando i loro comportamenti sul piano di controllo utilizzando un protocollo vettoriale di percorso ridotto (RPVP).

Questo modello consente una verifica efficiente degli stati convergenti senza dover simulare tutti gli stati intermedi. Inoltre, lo strumento integra ottimizzazioni come il già citato *pruning* basato sulle politiche, che limita l'esecuzione del protocollo solo alle parti della rete rilevanti per la politica da verificare (ad esempio, raggiungibilità, assenza di *loop*). Sebbene Plankton abbia ottenuto buoni risultati in molti casi, presenta dei limiti, come il supporto incompleto per l'instradamento BGP *multipath* e le difficoltà nel modellare le implementazioni di protocolli specifici di un *vendor*.

5.2.11.3 Diagnosi e Verifica delle Configurazioni di Rete

Il documento intitolato “*Using Service Grammar to Diagnose BGP Configuration Errors*” [98] illustra un metodo per diagnosticare gli errori di configurazione nei sistemi BGP attraverso un sistema chiamato appunto *Service Grammar*. Il presupposto è che i componenti della rete possono funzionare correttamente, ma i servizi *end-to-end* possono fallire a causa di errori di configurazione, errori che possono occorrere quando i parametri di configurazione sono impostati in modo errato con il rischio di causare interruzioni di servizio, anche se i singoli componenti della rete, come ad esempio i *router*, sono operativi. La caratteristica principale consiste nell'introduzione di un linguaggio dei requisiti, che contiene vincoli di alto livello derivati da ciò che può essere considerato una “configurazione corretta” per diversi protocolli. Tale linguaggio consente di comporre i requisiti a livello di rete, formando specifiche di architettura e di politica. Un motore di diagnosi verifica se i vincoli specificati sono validi per le configurazioni effettive del sistema. Se vengono rilevate incongruenze, il motore identifica gli errori e consente agli amministratori di regolare le configurazioni di conseguenza.

In riferimento al BGP, che può avere delle configurazioni complesse in ragione della flessibilità nel definire le politiche di instradamento per ciascun sistema autonomo, *Service Grammar* aiuta a definire i requisiti e le politiche specifiche della rete, verificando che siano correttamente implementati dalle configurazioni sui dispositivi. Quanto alla connettività, vengono introdotti requisiti come *iBGP session* ed *eBGP session*, che descrivono il successo

nella creazione rispettivamente di una sessione BGP tra *router* all'interno o all'esterno di un sistema autonomo. In merito alle politiche di instradamento poi, la grammatica consente di specificare regole globali a livello di AS, come la garanzia di percorsi di instradamento preferenziali, relazioni cliente-fornitore e filtraggio degli instradamenti in base alle politiche del singolo sistema autonomo. Quindi *Service Grammar* è in grado di diagnosticare sistematicamente gli errori attraverso la verifica di questi requisiti rispetto alle configurazioni effettive dei *router*.

Il documento intitolato “*Bagpipe: Verified BGP Configuration Checking*” [99] introduce un sistema chiamato *Bagpipe* per la verifica delle configurazioni del BGP in grado di assicurare che gli operatori di rete implementino correttamente le politiche di instradamento. Infatti consente agli ISP di esprimere in modo conciso le loro politiche di instradamento e di verificare automaticamente se le configurazioni dei loro *router* siano conformi a tali politiche. *Bagpipe* si basa sul *framework Rosette* che fa uso di strumenti come i solutori di soddisfacibilità (SAT solvers) per verificare le configurazioni; esegue una riduzione iniziale della rete (INR) per semplificare il problema della verifica di tutti i possibili stati dell'instradamento, tentando di ridurre il cosiddetto *overhead* computazionale; verifica parallelamente su più nodi, rendendo il processo scalabile per reti di grandi dimensioni.

5.2.11.4 Verifica Incrementale e Modifica delle Configurazioni

Il documento intitolato “*Incremental Network Configuration Verification*” [102] introduce un metodo per verificare in modo efficiente le modifiche alle configurazioni di rete in modo incrementale, anziché eseguire una nuova verifica completa dopo ogni modifica. Questo perché in effetti le configurazioni di rete possono cambiare spesso per supportare nuovi servizi, ospitare più dispositivi e migliorare la sicurezza. Sono modifiche che potenzialmente possono introdurre errori con conseguenti potenziali interruzioni della rete. Viene proposto INCV che dunque verifica, in modo efficiente, solo le parti della rete interessate da una modifica di configurazione. A differenza della verifica condotta sul piano dati (che comporta operazioni semplici come l'inoltro o lo scarto dei pacchetti), le modifiche alla configurazione sul piano di controllo (come sulle politiche di instradamento o l'aggiunta di adiacenze) sono più complesse e possono interessare più dispositivi. A questo punto è necessario determinare quali parti dello stato di instradamento e di inoltro rimangono valide dopo una modifica della configurazione e quali invece necessitano di una ulteriore verifica. I verificatori di

configurazione come *Batfish* simulano o analizzano i piani di controllo della rete, ma non sono progettati per la verifica incrementale.

In questo senso, il documento presenta *RealConfig*, un prototipo di sistema progettato per eseguire INCV attraverso la generazione solo degli aggiornamenti del piano dati necessari dopo le modifiche alla configurazione e mediante la verifica incrementale delle politiche di rete. *RealConfig* è costituito di tre componenti principali: un generatore incrementale del piano dati che crea gli aggiornamenti in base alle modifiche della configurazione grazie a *Differential Datalog* che evita di ricalcolare l'intero piano dati; un aggiornatore incrementale del modello del piano dati con le nuove regole di inoltro create dal generatore incrementale e con le modifiche elaborate in *batch* attraverso il contestuale aggiornamento delle classi di equivalenza (CE, definite come insiemi di pacchetti che condividono lo stesso comportamento di inoltro nella rete) interessate; un verificatore di politiche incrementali come la raggiungibilità e l'assenza di loop in base al modello del piano dati aggiornato (sono questi tre i componenti di INCV). Quest'ultimo componente verifica solo le politiche relative alle classi di equivalenza interessate rendendo di fatto più efficiente il controllo. Si tratta comunque di un lavoro che in futuro dovrà cimentarsi nella sperimentazione di configurazioni di rete reali e nell'esplorazione di metodi alternativi alla verifica incrementale.

Il documento intitolato “*A General Approach to Network Configuration Verification*” [85] presenta il già citato *Minesweeper*, uno strumento progettato per verificare che una configurazione di rete aderisca a un'ampia gamma di proprietà desiderate come la raggiungibilità, l'isolamento, il *waypointing* (cioè la verifica che il traffico transiti attraverso nodi specifici) la prevenzione dei buchi neri (cioè verifica dei casi in cui il traffico potrebbe essere interrotto involontariamente), il bilanciamento del carico e altro. Le reti tradizionali, a differenza delle SDN (*Software-Defined Networking*), hanno piani di controllo complessi che gestiscono dinamicamente l'instradamento e l'inoltro in base alle configurazioni. Queste configurazioni possono essere lunghe e complesse, cosa che può favorire errori e interruzioni. *Minesweeper* traduce le configurazioni di rete in una formula logica che modella lo stato stabile che si suppone la rete debba raggiungere quando il suo piano di controllo converge. Combinando la formula logica della rete con i vincoli che rappresentano le proprietà previste, lo strumento può verificare se esistono stati stabili nei quali la proprietà viene meno. Se è possibile soddisfare la formula, allora significa che esiste uno stato della rete in cui la proprietà non è valida, altrimenti la proprietà è garantita in tutti gli stati stabili.

Nel dettaglio il metodo verifica la raggiungibilità, cioè riesce a garantire che il traffico possa raggiungere una specifica destinazione; l'isolamento, cioè può confermare che due nodi non riescono a comunicare; il *waypointing*, cioè verifica che il traffico transiti attraverso nodi specifici; il rilevamento di buchi neri, cioè verifica dei casi in cui il traffico potrebbe essere interrotto involontariamente; la lunghezza del percorso limitata, cioè assicura che il traffico non segua un percorso inutilmente lungo; il bilanciamento del carico, cioè può garantire che il traffico sia distribuito su più percorsi; l'equivalenza funzionale, cioè verifica che due *router* si comportino in modo identico in tutti gli scenari; tolleranza ai guasti, cioè verifica che la rete possa tollerare guasti ai collegamenti o ai nodi senza perdere la connettività o violare altre proprietà. Tra i punti di forza di Minesweeper c'è il fatto che non analizza i percorsi dei singoli pacchetti, ma costruisce un modello a grafo che cattura tutte le interazioni tra i messaggi di instradamento attraverso la rete. È un metodo che consente di ragionare su un insieme di proprietà più ricco rispetto ai modelli tradizionali basati sui percorsi. Inoltre fa uso della ricerca combinatoria che consente di cercare direttamente una soluzione stabile che riesca a violare una proprietà, il che è più efficiente rispetto al calcolo di tutti i possibili stati di instradamento. Ciononostante, pur offrendo capacità di verifica complete, *Minesweeper* si limita a verificare le proprietà dopo che la rete ha raggiunto uno stato stabile, non durante stati transitori o instabili. Inoltre, potrebbe richiedere risorse aggiuntive per gestire reti con configurazioni di BGP interno complesse.

5.2.11.5 Verifica Formale e Tecniche di Model Checking

Un altro lavoro importante nel perimetro della verifica formale è “*NV: An Intermediate Language for Verification of Network Control Planes*” [113] scritto da Nick Giannarakis, Ryan Beckett, Devon Loehr e David Walker che tratta della creazione di un linguaggio intermedio chiamato NV. Ispirato ai linguaggi di verifica intermedi esistenti come *Boogie* e *Why3*, si concentra sull'espressività, garantendo al contempo la trattabilità dei compiti di verifica; include costrutti linguistici standard come interi, booleani, funzioni e record ed è progettato per rappresentare valori noti e sconosciuti (ad esempio, valori simbolici che rappresentano potenziali guasti della rete) e intende contribuire a superare le sfide associate alle errate configurazioni di rete. È stato progettato per modellare protocolli di instradamento come BGP definendo una funzione di trasferimento (come gli instradamenti vengono trasformati tra le reti) e una funzione di fusione (come i *router* selezionano il *best path*). Questo modello consente di analizzare le configurazioni di rete per verificare proprietà come

la raggiungibilità del piano di controllo e la tolleranza ai guasti. Il documento fornisce confronti che dimostrano la capacità di NV nel superare strumenti come *Batfish* e *MineSweeper* in termini di velocità e uso della memoria.

È presente nella letteratura scientifica un ulteriore lavoro su questo tema. Si tratta di “*Declarative Network Verification*” [103] di Wang, Basu, Loo e Sokolsky che tratta dello sviluppo e dell'uso di un verificatore di rete dichiarativo (DNV). Il DNV fa uso di una tecnica di verifica ben consolidata, il *theorem proving*, per catturare la semantica della rete in assiomi basati sulla logica: integra le specifiche di protocollo scritte in un linguaggio di interrogazione dichiarativo chiamato *Network Datalog* (NDlog) e traduce queste specifiche in assiomi logici per la verifica appunto mediante *theorem prover*. L'obiettivo è verificare i protocolli di rete (BGP compreso) senza richiedere un processo separato di specificazione del sistema, che di solito richiede tempo e sforzi considerevoli. La verifica comprende la dimostrazione dell'ottimalità del percorso e la gestione dei cambiamenti dinamici, come i guasti ai collegamenti di rete. Il documento inoltre spiega come vengano trattati gli aggiornamenti *soft-state* nei protocolli di rete, cioè quelle informazioni che nella rete scadono dopo un certo periodo di tempo e devono essere aggiornate. Questo aspetto è fondamentale per le reti del mondo reale, dove gli stati transitori e i guasti sono comuni.

5.2.11.6 Verifica e Monitoraggio delle Reti Dinamiche

Il documento intitolato “*Checking Beliefs in Dynamic Networks*” [104] esplora un metodo per verificare le configurazioni di rete attraverso uno strumento chiamato *Network Optimized Datalog* e si concentra sulla verifica delle credenze che gli operatori di rete hanno sulla configurazione e sul comportamento delle reti dinamiche, come le politiche relative alla raggiungibilità della rete, alla sicurezza e alla tolleranza ai guasti. Una delle credenze esposte nel lavoro si riferisce a un invariante di alto livello che gli operatori di rete ritengono vera sulla rete, come per esempio “I *controller* interni non sono accessibili da Internet” o “Il traffico dalle macchine virtuali *guest* non può raggiungere i *server* di gestione”. In questi casi NoD aiuta a verificare tali convinzioni modellando la rete con Datalog, che può codificare sia gli stati della rete che le politiche. A quel punto, se una convinzione è falsa a causa di errori di configurazione o di eccezioni alle politiche, NoD può scoprire i problemi e aiutare a perfezionare le ipotesi dell'operatore. Supporta anche le reti dinamiche, dove le intestazioni dei pacchetti, i formati o i comportamenti di inoltro possono cambiare grazie a tecnologie come OpenFlow [105] o P4 [106]. NoD fornisce una verifica più veloce e scalabile rispetto

agli strumenti tradizionali come i solutori SAT e i *model checker*, soprattutto quando si tratta di reti dinamiche o su larga scala. Va detto anche che NoD non è limitato alla verifica di un insieme fisso di proprietà statiche o predefinite. È uno strumento flessibile che supporta la personalizzazione e la verifica di proprietà dinamiche e definite dagli operatori.

Sul fronte del piano dati, il documento intitolato “*Automatically Verifying Reachability and Well-formedness in P4 Networks*” [107] illustra le sfide e le soluzioni per verificare la correttezza e la sicurezza del comportamento della rete nei sistemi che utilizzano P4, un linguaggio di programmazione flessibile per i piani dati nelle reti *software-defined* (SDN) che consente di modificare al volo le intestazioni e i comportamenti di inoltramento. Tale flessibilità, tuttavia, introduce anche un rischio più elevato di errori *software*, ad esempio nell’inoltramento e nella eventuale presenza di pacchetti malformati. Gli autori propongono un metodo automatico per verificare sia la raggiungibilità che la correttezza delle reti P4. Il metodo prevede la compilazione dei programmi P4 in Datalog, un linguaggio di programmazione logica che consente l’aggiornamento automatico del modello di verifica quando la rete cambia cosa che consente una verifica continua senza interventi manuali. Il sistema di verifica consiste nel controllare le proprietà della rete come raggiungibilità e la corretta formattazione dei pacchetti. Esegue anche *test* di regressione per garantire che le riconfigurazioni della rete non introducano nuovi *bug*. Il lavoro reca anche esempi del sistema per rilevare errori in vari casi d’uso, come il bilanciamento del carico con il protocollo *mTag* e l’applicazione dei criteri di sicurezza con il protocollo *sTag*. Il processo di verifica assicura che qualsiasi modifica apportata alla logica di instradamento della rete, come l’aggiunta di nuovi *tag* di sicurezza, non introduca errori o violazioni del protocollo.

Di seguito una sinossi che espone, per ogni tipo di attacco citato, quale sia la possibile soluzione o misura di difesa. Nell’ultima colonna a destra c’è il riferimento, ove presente, ai documenti sulla verifica formale esaminati:

Attacco BGP	Descrizione	Soluzione/Misura di difesa	Lavoro di riferimento della verifica formale
Dirottamento degli	Annuncio di reti	S-BGP, BGPsec,	

instradamenti	non autorizzate che dirotta il traffico verso l'AS sbagliato	RPKI, ARTEMIS, BGP Monitoring e Visualizzazione	
Instabilità degli Instradamenti e del BGP	Annunci e ritiri rapidi e continui di instradamenti, che causano instabilità della rete	GTSM, TCP-AO, RTBH, Flowspec, Scrubbing Center	
Fuga di Instradamenti	Propagazione di instradamenti non autorizzati a terze parti	GTSM, TCP-AO, RTBH, Flowspec, Scrubbing Center	
Disaggregazione dei Prefissi BGP	Annuncio di reti più piccole invece di un aggregato più grande che causa sovraccarico dei dispositivi	BGPsec, Flowspec, Scrubbing Center, Verifica Formale	Abstract Interpretation of Distributed Network Control Planes', Ryan Beckett et al., 2020; NV: An Intermediate Language for Verification of Network Control Planes, Nick Giannarakis et al., 2020
Dirottamento della Sessione BGP	Compromissione di una sessione BGP	TCP-AO, GTSM, MD5 su sessioni	Bagpipe: Verified BGP Configuration

	per alterare o deviare il traffico	BGP, Verifica Formale	Checking, 2020; Fast BGP Simulation of Large Datacenters, 2020
Manipolazione delle Politiche BGP	Manipolazione delle politiche di instradamento per influenzare percorsi a livello globale	Service Grammar, Verifica Formale, BGP Monitoring	Using Service Grammar to Diagnose BGP Configuration Errors, 2020 Declarative Network Verification, Wang et al., 2020

5.2.12 VERIFICA FORMALE: LIMITI

La verifica formale, compresa l'astrazione, specialmente su larga scala, può richiedere un'enorme quantità di risorse computazionali, rendendola poco praticabile per reti molto grandi. E inoltre, nonostante l'innegabile utilità per garantire la correttezza teorica, la verifica formale non offre una protezione attiva contro attacchi in tempo reale. Infine, per reti di grandi dimensioni, implementare verifiche formali su ogni *router* può essere logisticamente difficile e richiedere strumenti sofisticati. Ecco di seguito una concisa antologia dei limiti dell'approccio formale per gli ambienti di produzione degli operatori di rete.

5.2.12.1 Complessità computazionale

La verifica formale, specialmente su larga scala, richiede un'enorme quantità di risorse computazionali. Nonostante le ottimizzazioni, strumenti come Minesweeper, Plankton o NV possono incontrare difficoltà a gestire reti di grandi dimensioni con configurazioni complesse, come quelle dei sistemi autonomi che implementano sia internal sia external BGP.

5.2.12.2 Scalabilità

Le tecniche di verifica formale, come il *model checking* o il *theorem proving*, devono esplorare spazi di stato molto ampi che aumentano esponenzialmente con la complessità delle reti e delle politiche di instradamento. Anche strumenti ottimizzati come Plankton e Minesweeper soffrono in reti molto estese o dinamiche.

5.2.12.3 Stato transitorio

La maggior parte degli strumenti di verifica si concentra su stati stabili del piano di controllo, ignorando i comportamenti durante gli stati transitori. Le eventuali anomalie o vulnerabilità che emergono durante la convergenza del protocollo non vengono rilevate.

5.2.12.4 Mancanza di protezione in tempo reale

La verifica formale garantisce la correttezza teorica delle configurazioni, ma non fornisce una protezione attiva contro gli attacchi in tempo reale, come il BGP hijacking. Strumenti come ARTEMIS sono più adatti al rilevamento e alla mitigazione degli attacchi in tempo reale.

5.2.12.5 Difficoltà logistiche

Applicare la verifica formale a tutte le configurazioni dei *router* in una rete globale può essere logisticamente complesso, richiedere infrastrutture sofisticate e personale altamente specializzato.

5.2.12.6 Dipendenza da modelli e specifiche

Gli strumenti di verifica dipendono da modelli accurati delle configurazioni e dei protocolli che, tuttavia, potrebbero non riflettere le complessità o le peculiarità delle implementazioni reali. Ad esempio, strumenti come Batfish possono incontrare difficoltà nel modellare comportamenti non documentati o *bug* specifici di un fornitore.

5.2.12.7 Difficoltà con le reti dinamiche

Le reti che usano tecnologie dinamiche, come SDN (*Software Defined Networking*) o linguaggi programmabili come il P4, introducono modifiche rapide e imprevedibili che complicano l'applicazione della verifica formale. Strumenti come NoD (*Network Optimized Datalog*) offrono alcune soluzioni, ma non sono esenti da limitazioni.

5.2.12.8 Limitazioni nei tipi di verifica

Non tutti gli strumenti coprono l'intero spettro delle proprietà di rete: alcuni infatti si concentrano su proprietà statiche, come la raggiungibilità e l'assenza di *loop*. Altri, come Plankton, non supportano completamente funzionalità come l'instradamento *multipath* di BGP.

5.2.13 CONSIDERAZIONI CONCLUSIVE

Gli studi e le riflessioni strettamente connessi a questo argomento si protraggono da vent'anni circa: alcuni metodi di verifica hanno avuto una certa fortuna in ambito accademico e quindi sono stati ripresi, migliorati, ampliati. Tuttavia sembra che questi sforzi non siano riusciti finora ad abbattere il muro di resistenza dell'industria che non ha incorporato, né tantomeno recepito, tali metodi in nessun prodotto (né nei sistemi operativi, né nei software di gestione). E avrebbe potuto farlo. Il punto debole risiederebbe nella necessità di onnicomprensività di un metodo in grado di valutare allo stesso tempo tutte le sfaccettature delle complesse configurazioni dei *router*: sessioni BGP (IPv4 e IPv6) di tipo interno ed esterno (anche *multihop*), tempi di convergenza, filtri, liste per i controlli di accesso, ecc. Valutare inoltre le informazioni contenute negli IRR confrontando quei dati con quanto rilevato dai *route collector* così da poter verificare, ad esempio, la reale funzionalità di filtri e sessioni BGP.

Oggi dunque l'industria esprime il bisogno di uno strumento semplice capace di aiutare gli operatori di rete a verificare le configurazioni dei propri *router* affinché siano conformi a degli standard riconosciuti di sicurezza, a delle BCOP (*Best Current Operational Practice*) codificate. Praticamente è necessario un metodo in grado di certificare che gli operatori di rete possano superare i *test* suggeriti da enti internazionali in merito all'igiene e alla sicurezza degli instradamenti su Internet. Indicativo è il brano inserito nella norma europea NIS2 [\[79\]](#) relativo proprio a questo aspetto:

“applicare le migliori pratiche per la sicurezza del DNS e per la sicurezza e l'igiene dell'instradamento in internet del traffico proveniente dalla rete e ad essa destinato”.

Per questi motivi, il presente lavoro di ricerca si è indirizzato a riempire questa lacuna, attraverso lo sviluppo dello strumento chiamato RoSe-T, cioè *Routing Security Tool*, basato su un metodo di emulazione che soddisfi i requisiti dell'iniziativa MANRS, *Mutually Agreed Norms for Routing Security*, della quale si parlerà diffusamente nel prossimo capitolo.

6 *MUTUALLY AGREED NORMS FOR ROUTING SECURITY*

Nel corso degli anni, diverse iniziative si sono occupate di raccogliere le migliori pratiche nell'ambito della sicurezza del *routing* e molto è stato fatto in termini di promozione di una cultura di responsabilità collettiva verso la sicurezza stessa dell'ecosistema. Tra le varie iniziative è necessario illustrare la più importante e la più longeva, ovvero MANRS (*Mutually Agreed Norms for Routing Security*).



MANRS

Nata nel 2014 [\[67\]](#), promossa e sostenuta inizialmente da ISOC, *Internet Society*, MANRS è un'iniziativa globale (dal 2024 passata sotto l'egida di GCA, *Global Cyber Alliance* [\[68\]](#)) che fornisce alcune soluzioni efficaci per la riduzione e la mitigazione delle principali minacce all'instradamento su Internet. MANRS viene costantemente aggiornato, seguendo l'evoluzione dei meccanismi di difesa dalle minacce di *routing* più comuni, ed è quindi una guida per tutti quegli operatori che vogliano migliorare e mantenere aggiornato il livello di sicurezza della propria infrastruttura. Fornisce programmi diversi in base al tipo di rete e alle attività svolte nell'ecosistema. Le azioni suggerite riguardano gli *Internet Exchange Point*, i *Content Provider* e le *Content Delivery Network*, i fornitori e soprattutto gli operatori di rete. L'idea di base è quella di:

- aumentare la consapevolezza dei problemi di sicurezza del *routing* e incoraggiare l'implementazione di azioni in grado di risolverli;
- promuovere una cultura di responsabilità collettiva verso la sicurezza;
- fornire agli operatori un quadro di riferimento per comprendere e affrontare meglio i problemi.

6.1 Formalismo di specifica e linguaggio di implementazione

Al fine di illustrare e confermare le misure raccomandate dal programma MANRS in merito alla sicurezza del *routing*, si adotta un formalismo basato su predicati logici e clausole di

regola, utili a esprimere obblighi, permessi e divieti in maniera chiara e formalizzata. Tale approccio metodologico consente di rappresentare in forma simbolica le relazioni tra operatori, prefissi IP, pacchetti e sessioni BGP.

Il formalismo di specifica impiegato è una logica dei predicati di primo ordine arricchita da costrutti che distinguono tra:

- relazioni tra entità ($\text{customer}(\text{op}, \text{cust})$, $\text{ownPrefix}(\text{op}, \text{prefix})$);
- eventi ($\text{announcement}(\text{op}, \text{neighbor}, \text{prefix})$);
- condizioni ($\text{verified}(\text{op}, \text{cust}, \text{prefix})$);
- vincoli ($\rightarrow, \neg, \exists, \forall$) che esprimono obblighi o divieti operativi.

Tali regole vengono implementate in un linguaggio dichiarativo, il quale si ispira alla sintassi e semantica del *Prolog*, particolarmente adatto alla rappresentazione di sistemi basati su conoscenza e all'esecuzione di verifiche logiche su insiemi di fatti e regole. Nello specifico, vengono utilizzate clausole di Horn e *predicate logic* per modellare il comportamento atteso degli operatori in conformità con le *policy* di sicurezza.

Questo formalismo consente non solo una descrizione rigorosa, ma anche l'automatizzazione della verifica di conformità alle specifiche MANRS, rendendo possibile l'analisi di configurazioni BGP e l'identificazione di comportamenti non conformi. Comunque è necessario precisare che nel presente capitolo si usano due livelli di formalizzazione:

- logica dei predicati per esprimere obblighi e vincoli generali;
- clausole dichiarative (in stile *Prolog*) per l'implementazione e la verifica automatica.

Inoltre, nel formalismo logico adottato per rappresentare le azioni del programma MANRS, si assume la *Closed World Assumption* (CWA), ovvero l'ipotesi del mondo chiuso. Secondo tale assunzione, tutti i fatti che non sono esplicitamente affermati nel sistema sono considerati falsi. Questo approccio si contrappone alla *Open World Assumption* (OWA), dove un fatto non noto viene trattato come "sconosciuto". L'adozione della CWA è tipica nei sistemi a logica dichiarativa come *Prolog*, ed è particolarmente adatta per modellare *policy* di controllo, come quelle sulla sicurezza del *routing*, in cui l'assenza di autorizzazione o di verifica costituisce una violazione. Infine, tutti i predicati qui usati sono stati documentati nell'[Appendice 11](#).

Tornando a MANRS, il manifesto specifica una serie di azioni importanti che mirano ad affrontare essenzialmente tre tipi di problemi:

- informazioni di raggiungibilità non corrette;
- traffico con indirizzi IP di origine *spoofed*;
- coordinamento e collaborazione tra gli operatori.

Ecco infine le quattro azioni definite dall'iniziativa per gli operatori di rete.

6.1.1 AZIONE 1

6.1.1.1 Impedire la propagazione di informazioni errate di instradamento

Prevenire la propagazione di instradamenti errati: è necessario implementare un sistema sulla base del quale l'operatore annunci ai propri dirimpettai solo ed esclusivamente i propri prefissi e quelli dei propri clienti. A sua volta l'operatore deve verificare che i prefissi annunciati dai propri clienti siano effettivamente i loro.

6.1.1.2 Relazioni e predicati

- `operator(op)`: esiste l'operatore `op`. Questo significa che `op` è un'entità riconosciuta nel sistema, in questo caso un operatore di rete.
- `customer(op, cust)`: l'operatore `op` ha un cliente `cust`. Questo predicato indica una relazione di clientela tra `op` e `cust`.
- `ownPrefix(op, prefix)`: il prefisso `prefix` appartiene all'operatore `op`. `op` possiede il prefisso IP specificato.
- `customerPrefix(cust, prefix)`: il prefisso `prefix` appartiene al cliente `cust`. Questo indica che l'operatore ha controllato e confermato che il prefisso del cliente è valido
- `neighbor(op, neighbor)`: `neighbor` è un altro operatore di rete rispetto a `op`. Entrambi stabiliscono una sessione BGP fra loro.
- `announcement(neighbor, prefix)`: l'operatore `neighbor` annuncia un prefisso.

6.1.1.3 Regole e clausole

Queste regole definiscono obblighi, permessi e proibizioni legati agli annunci di prefissi da parte degli operatori. La seguente espressione non è una clausola nel senso dei linguaggi

logici come *Prolog*, bensì una formula in logica dei predicati che esprime un vincolo obbligatorio nel sistema:

Obbligo: $\forall op, neighbor, prefix: announcement(op, neighbor, prefix) \rightarrow (ownPrefix(op, prefix) \vee \exists cust: customer(op, cust) \wedge verified(op, cust, prefix))$

Ogni volta che un operatore *op* annuncia un prefisso *prefix* a un dirimpettaio *neighbor*, deve essere vero che:

- l'operatore *op* possiede quel prefisso oppure
- esiste un cliente *cust* dell'operatore che possiede il prefisso e l'operatore ha verificato che quel prefisso appartiene al cliente.

Permesso: $\forall op, neighbor, prefix: announcement(op, neighbor, prefix) \rightarrow ownPrefix(op, prefix)$

L'operatore *op* è autorizzato ad annunciare il prefisso *prefix* al dirimpettaio *neighbor* se possiede il prefisso. Questo predicato rappresenta una condizione in cui l'annuncio è considerato valido: cioè se un operatore possiede il prefisso e lo annuncia, tale annuncio è lecito. Si tratta quindi di una condizione sufficiente, non necessaria, per la validità dell'annuncio.

Permesso: $\forall op, neighbor, cust, prefix: announcement(op, neighbor, prefix) \rightarrow (customer(op, cust) \wedge verified(op, cust, prefix))$

L'operatore *op* è autorizzato ad annunciare un prefisso al dirimpettaio *neighbor* se:

- il prefisso appartiene a un cliente *cust* dell'operatore,
- l'operatore ha verificato che il prefisso appartiene al cliente.

Infine, nel sistema logico adottato, che si basa sull'assunzione del mondo chiuso, è possibile esprimere in modo formale la proibizione, prevista dalle regole MANRS, per cui a un operatore *op* è vietato annunciare un prefisso *prefix* qualora non sia né di sua proprietà, né verificato come appartenente a un proprio cliente.

6.1.2 AZIONE 2

6.1.2.1 Bloccare il traffico con indirizzi IP di origine falsificati

Un operatore dovrebbe implementare un sistema in grado di operare la Source Address Validation per la propria rete e per i propri clienti. È quindi necessario che attivi filtri cosiddetti *anti-spoofing* per evitare che pacchetti con un IP sorgente errato entrino o escano dalla sua rete.

6.1.2.2 Relazioni e predicati

- `operator(op)`: l'operatore `op` gestisce una rete. Questa affermazione dichiara che `op` è un gestore di una rete.
- `customer(op, cust)`: l'operatore `op` ha il cliente `cust`, indicando una relazione tra operatore e cliente.
- `network(op, net)`: l'operatore `op` gestisce la rete `net`.
- `sourceAddressValidation(net)`: la rete `net` ha implementato la Source Address Validation (SAV), ovvero un meccanismo che assicura che gli indirizzi IP di origine nei pacchetti siano validi per quella rete.
- `antiSpoofingFilter(net)`: la rete `net` ha implementato filtri anti-spoofing, per prevenire che pacchetti con indirizzi di origine falsificati vengano inoltrati.
- `packet(src, dst, data)`: un pacchetto ha un indirizzo sorgente `src`, un indirizzo di destinazione `dst` e dati `data`.
- `correctSource(packet, net)`: l'indirizzo sorgente `src` del pacchetto è corretto e valido per la rete `net`, ovvero è un indirizzo di origine autorizzato per quella rete.

6.1.2.3 Regole e clausole

Obbligo: $\forall op, net: operator(op) \wedge network(op, net) \rightarrow sourceAddressValidation(net)$

Questa regola impone un obbligo: ogni operatore `op` che gestisce una rete `net` deve implementare la Source Address Validation per quella rete. Effettivamente, tutte le reti gestite dall'operatore devono avere attivi i meccanismi di convalida dell'indirizzo sorgente.

Obbligo: $\forall op, net: operator(op) \wedge network(op, net) \rightarrow antiSpoofingFilter(net)$

Un secondo obbligo: ogni operatore *op* che gestisce una rete *net* deve implementare i filtri anti-spoofing per quella rete. Ciò significa che la rete deve essere configurata per bloccare pacchetti che utilizzano indirizzi sorgente falsificati.

Permesso: $\forall op, net, packet: operator(op) \wedge network(op, net) \wedge correctSource(packet, net) \rightarrow allow(packet, net)$

Questa affermazione specifica che l'operatore ha il permesso di consentire il transito di un pacchetto sulla sua rete, a condizione che:

- l'operatore *op* gestisca la rete *net* (attraverso il predicato *network*(*op*, *net*)), e
- il pacchetto *packet* abbia un indirizzo sorgente corretto e valido per la rete (*correctSource*(*packet*, *net*)).

Dunque, un pacchetto con un indirizzo di origine valido può essere consentito sulla rete. Anche in questo caso, nel sistema logico adottato, che si basa sull'assunzione del mondo chiuso, è possibile esprimere in modo formale la proibizione, prevista dalle regole MANRS, per cui ogni pacchetto che non ha un indirizzo di origine valido per la rete ($\neg correctSource(packet, net)$) deve essere bloccato dalla rete (*deny*(*packet*, *net*)).

6.1.3 AZIONE 3

6.1.3.1 Facilitare il coordinamento e la collaborazione con altri operatori

Le informazioni di contatto (come e-mail e numeri di telefono del NOC, *Network Operations Centre*) all'interno degli IRR e di peeringDB devono essere mantenute aggiornate.

6.1.3.2 Relazioni e predicati

- *IRR*(*irr*): esiste un *database* IRR. Questa affermazione stabilisce che esiste un *database* IRR, che contiene informazioni di *routing* utilizzate dagli operatori di rete.
- *peeringDB*(*pdb*): esiste il *database* PeeringDB, che fornisce informazioni su come e dove gli operatori di rete possono effettuare *peering*.
- *contactInfo*(*entity*, *irr/pdb*, *info*): l'entità *entity* (un operatore di rete) ha delle informazioni di contatto presenti nel *database* IRR o PeeringDB, rappresentate da *info*.

- $upToDate(info, time)$: le informazioni $info$ sono aggiornate al momento specifico $time$.
- $currentTime(T)$: T rappresenta l'ora corrente.

6.1.3.3 Regole e clausole

Obbligo: $\forall irr, entity, info : IRR(irr) \wedge$
 $contactInfo(entity, irr, info) \wedge currentTime(T) \wedge$
 $\neg upToDate(info, T) \rightarrow MustUpdate(info, T)$

Questa regola impone un obbligo: se un'entità (un operatore di rete) ha delle informazioni di contatto registrate nel *database* IRR ($contactInfo(entity, irr, info)$) e queste informazioni non sono aggiornate rispetto all'ora corrente ($\neg upToDate(info, T)$), allora tali informazioni devono essere considerate come necessarie da aggiornare ($MustUpdate(info, T)$).

In altre parole, gli operatori di rete hanno l'obbligo di aggiornare le loro informazioni di contatto nel *database* IRR ogni volta che queste non sono più aggiornate.

Obbligo: $\forall pdb, entity, info : peeringDB(pdb) \wedge$
 $contactInfo(entity, pdb, info) \wedge \neg upToDate(info,$
 $currentTime()) \rightarrow update(info, currentTime())$

Analogamente alla prima regola, questa regola riguarda il *database* PeeringDB. Se un'entità (operatore di rete) ha delle informazioni di contatto presenti nel *database* PeeringDB ($contactInfo(entity, pdb, info)$) e queste non sono aggiornate rispetto all'ora corrente ($\neg upToDate(info, currentTime())$), l'entità ha l'obbligo di aggiornare queste informazioni ($update(info, currentTime())$).

Quindi, gli operatori devono garantire che le informazioni di contatto nel PeeringDB siano aggiornate nel caso in cui risultino obsolete.

6.1.4 AZIONE 4

6.1.4.1 Documentare le proprie informazioni di instradamento

Questo passaggio è necessario per facilitare ai dirimettai BGP di allestire il filtraggio. Dunque le informazioni di raggiungibilità devono essere riportate sia all'interno degli IRR

(tramite gli oggetti *route:* e *route6:*) sia nell'elenco dei *downstream* (nell'oggetto *as-set:*), e creare ROA per i prefissi.

6.1.4.2 Relazioni e predicati

- $IRR(irr)$: esiste un *database* IRR. Questa affermazione stabilisce l'esistenza di un *database* IRR che contiene informazioni di *routing*, come i prefissi IP e gli oggetti associati.
- $routeObject(irr, prefix, info)$: il *database* IRR *irr* contiene un oggetto *route* per il prefisso *prefix* con le informazioni *info*. Questo oggetto rappresenta il prefisso IPv4 associato a un determinato AS (sistema autonomo).
- $route6Object(irr, prefix, info)$: il *database* IRR *irr* contiene un oggetto *route6* per il prefisso *prefix* con le informazioni *info*. Questo è l'equivalente per IPv6 dell'oggetto *route:* per IPv4.
- $asSetObject(irr, entity, prefixes)$: il *database* IRR *irr* contiene un oggetto *as-set* per l'entità *entity*, che include un insieme di prefissi IP associati a quell'operatore. Gli *as-set* sono utilizzati per raggruppare prefissi e fornire informazioni sulle politiche di *routing*.
- $ROA(prefix, entity)$: esiste un ROA (*Route Origin Authorization*) per il prefisso *prefix* che autorizza l'entità (operatore) a pubblicizzare quel prefisso.
- $BGPNeighbor(entity, neighbor)$: l'entità (operatore) *entity* ha come dirimettaio BGP l'identificativo *neighbor*. Questo rappresenta la relazione tra un operatore e un vicino BGP.
- $filterable(info, neighbor)$: le informazioni *info* sono filtrabili per il vicino BGP *neighbor*. Questo indica che il vicino BGP può filtrare o bloccare le informazioni di *routing* se non sono appropriate o complete.

6.1.4.3 Regole e clausole

Obbligo: $\forall entity, neighbor, prefix: BGPNeighbor(entity, neighbor) \wedge \neg \exists irr, info: routeObject(irr, prefix, info) \wedge \neg \exists irr, info: route6Object(irr, prefix, info): filterable(info, neighbor) \rightarrow createRouteObject(entity, irr, prefix, info)$

Questa regola impone un obbligo: se un'entità *entity* ha un vicino BGP *neighbor* ($\text{BGPNeighbor}(\text{entity}, \text{neighbor})$) e non esiste un oggetto *route* o *route6* per il prefisso in nessun *database* IRR ($\neg \exists \text{ irr}: \text{routeObject}(\text{irr}, \text{prefix}, \text{info})$ e $\neg \exists \text{ irr}: \text{route6Object}(\text{irr}, \text{prefix}, \text{info})$), allora:

- se le informazioni *info* sono filtrabili per il vicino BGP ($\text{filterable}(\text{info}, \text{neighbor})$), l'entità deve creare un oggetto *route* nel *database* IRR ($\text{createRouteObject}(\text{entity}, \text{irr}, \text{prefix}, \text{info})$).

Cioè, se le informazioni di *routing* non sono presenti nei *database* IRR e sono soggette a filtri da parte di un vicino BGP, l'operatore ha l'obbligo di registrare correttamente il prefisso creando un oggetto *route* nel *database* IRR.

Obbligo: $\forall \text{ entity}, \text{ neighbor}, \text{ prefixes} : \text{BGPNeighbor}(\text{entity}, \text{neighbor}) \wedge \text{ownPrefix}(\text{entity}, \text{prefixes}) \wedge \neg \exists \text{ irr} : \text{asSetObject}(\text{irr}, \text{entity}, \text{prefixes}) \rightarrow \text{createAsSetObject}(\text{irr}, \text{entity}, \text{prefixes})$

Questa regola impone un altro obbligo: se l'entità (operatore) *entity* ha un vicino BGP *neighbor* ($\text{BGPNeighbor}(\text{entity}, \text{neighbor})$) e possiede un insieme di prefissi associati ($\text{ownPrefix}(\text{entity}, \text{prefixes})$), ma non esiste un oggetto *as-set* associato all'entità per tali prefissi in alcun *database* IRR ($\neg \exists \text{ irr}: \text{asSetObject}(\text{irr}, \text{entity}, \text{prefixes})$), allora l'entità (operatore) ha l'obbligo di creare un oggetto *as-set* nel *database* IRR ($\text{createAsSetObject}(\text{irr}, \text{entity}, \text{prefixes})$).

Praticamente, se l'operatore non ha ancora registrato un *as-set*: con i prefissi che gestisce, ha l'obbligo di crearne uno nel *database* IRR.

6.2 RELAZIONI TRA SISTEMI AUTONOMI

Al fine di inquadrare più approfonditamente il contesto del presente studio, è essenziale addentrarsi nei meandri delle relazioni che intercorrono tra i sistemi autonomi. È un aspetto così dirimente per la sicurezza di Internet che anche IETF [\[69\]](#), nella RFC 9234, si è recentemente pronunciato sul punto andando a definire i cosiddetti RUOLI del BGP.

6.2.1 RUOLI BGP

Provider, Customer, Route-Server, Route-Server Client, Peer: sono questi cinque i ruoli definiti nel testo della RFC 9234. Cioè, a prescindere dai rapporti economici che intercorrono tra i sistemi autonomi, è possibile classificare ciascun protagonista del mondo BGP, sessione per sessione, in uno dei ruoli citati. Sessione per sessione significa che un AS può essere classificato come *Customer* in un rapporto BGP, ma come *Provider* in un altro e così via. Questa tassonomia è utile per prevenire, a livello di configurazione, errati annunci di prefissi. Infatti un *Provider* può annunciare qualunque instradamento ai clienti; un *Customer* può annunciare a un *Provider* solamente qualunque instradamento appreso da un cliente (opportunamente filtrato) o originato localmente; un *Route-Server* può annunciare qualunque instradamento ai *Route-Server Client*; un *Route-Server Client* può annunciare a un *Route-Server* solamente qualunque instradamento appreso da un cliente (anche in questo caso, opportunamente filtrato) o originato localmente; un *Peer* può annunciare a un altro *peer* solamente qualunque instradamento appreso da un cliente o originato localmente.

Si comprende bene che se un *Peer* riceve da un suo *peer* un instradamento che quest'ultimo apprende dal *Provider*, tale instradamento va scartato poiché non appartiene né a un cliente né è originato localmente. La misura se applicata scrupolosamente è valida per ridurre al massimo i cosiddetti *route leak*. Ma è possibile fare anche meglio e lo prevede la stessa RFC 9234 quando introduce l'attributo BGP OTC (*Only To Customer*) che si abbina a un instradamento e ne designa la destinazione. Se infatti un instradamento OTC viene inviato a un *Customer*, a un *Peer* o a un *Route-Server Client*, l'attributo fa in modo che venga inoltrato a sua volta solo a *neighbor* classificati con il ruolo di *Customer*. Troppo bello per essere vero, infatti c'è un grosso ma: non tutti i *vendor* hanno implementato i ruoli BGP, per questo è possibile configurarli solo là dove le piattaforme lo consentano. A ogni modo, le principali relazioni che intercorrono tra i sistemi autonomi, anche alla luce della citata RFC, possono essere classificate in tre tipi diversi: fornitore-cliente, cliente-fornitore, e *peer*. Per un sistema autonomo è assai comune instaurare relazioni BGP contemporaneamente con molti altri sistemi autonomi. Così per esempio, come detto, un AS può stabilire molte o addirittura moltissime sessioni BGP con altri AS alla pari, *peering*, e al contempo mantenere due o più relazioni di transito (cioè con due o più fornitori), mentre eroga connettività a un certo numero di clienti. A questo punto, al fine di verificare se la configurazione BGP di ciascun AS è conforme alle azioni stabilite dal programma MANRS (segnatamente alla prima e alla

seconda), è necessario comprendere, per ogni sessione BGP, quale sia la relazione che intercorre tra i due sistemi autonomi coinvolti.

6.2.2 FORNITORE-CLIENTE

Il sistema autonomo AS1 (il fornitore) annuncia prefissi, appartenenti a tutti gli altri AS che conosce, al sistema autonomo AS2 (il cliente). Per ogni prefisso IP p annunciato da tutti gli altri AS di Internet (FIRT) esiste almeno un instradamento $(rp, 1)$ nella tabella di *routing* $T1$ di AS1 che conduce a quel prefisso; allo stesso tempo esiste un instradamento $(rp, 2)$ nella tabella di *routing* $T2$ di AS2 che conduce ad AS1.

Formalismo logico

$$\forall p \in \text{FIRT} : \exists rp,1 \in T1 \wedge \exists rp,2 \in T2$$

Implementazione *Prolog*

```
% Un prefisso P fa parte della FIRT
firt(P).

% AS1 ha un percorso Rp1 verso il prefisso P
route(as1, P, Rp1) :-
    firt(P),
    compute_route(as1, P, Rp1).

% AS2 ha un percorso Rp2 verso P costruito da quello di AS1
route(as2, P, [as2 | Rp1]) :-
    route(as1, P, Rp1).
```

Se un prefisso P fa parte dell'insieme di prefissi noti in Internet ($\text{firt}(P)$), allora:

- esiste un instradamento $Rp1$ nella tabella di *routing* di $As1$ per il prefisso P ;
- esiste un instradamento $Rp2$ nella tabella di *routing* di $As2$ per il prefisso P ;
- l'instradamento $Rp2$ è costruito aggiungendo $As2$ come primo nodo al percorso $Rp1$.

Il predicato `route/3` rappresenta una relazione tra: un sistema autonomo (AS) specifico, un prefisso IP, un percorso verso quel prefisso.

Verifica della condizione di fornitore

Perché un sistema autonomo *As1* sia considerato un *provider*, deve possedere un percorso verso ogni prefisso noto su Internet, ovvero ogni prefisso appartenente all'insieme *FIRT*.

```
% As1 non è un provider se esiste almeno un prefisso P nella
FIRT
% per cui non ha alcun percorso noto.
not_provider(As1) :-
    firt(P),
    \+ route(As1, P, _).
```

Il predicato `not_provider/1` è vero se esiste almeno un prefisso *P* per cui *As1* non ha un percorso valido. Di conseguenza, affinché *As1* sia considerato un *provider*, deve avere un percorso per tutti i prefissi in *FIRT*.

In termini logici, questo equivale a negare l'obbligo implicito:

$$\forall p \in \text{FIRT} \rightarrow \exists rp_1: \text{route}(\text{As1}, p, rp_1)$$

Nel caso in cui *As1* sia un *provider* di *As2*, ogni prefisso IP raggiungibile da *As1* deve essere raggiungibile anche da *As2*, il quale apprende i percorsi da *As1* e li modifica antepoendo se stesso. Cioè a dire::

Formalismo logico

$$\forall P \in \text{FIRT}: \text{route}(\text{As1}, P, \text{Rp}_1) \rightarrow \text{route}(\text{As2}, P, \text{Rp}_2) \wedge \text{Rp}_2 \\ = [\text{As2} \mid \text{Rp}_1]$$

Se *As1* ha un percorso *Rp1* verso il prefisso *P*, allora *As2* deve avere un percorso *Rp2* costruito antepoendo *As2* al percorso appreso da *As1*. Questo riflette il comportamento tipico BGP nella relazione cliente-fornitore.

Implementazione in *Prolog*

```
% As2 apprende i percorsi da As1 antepoendo se stesso
route(as2, P, Rp2) :-
    route(as1, P, Rp1),
    Rp2 = [as2 | Rp1].
```

Nel formalismo adottato, si usano due varianti del predicato `route`:

- `route/3` rappresenta la presenza di un percorso verso un prefisso nella tabella di *routing* di un AS, nella forma `route(AS, Prefix, Route)`;
- `route/4` modella invece in modo esplicito la relazione tra due sistemi autonomi in una sessione BGP, nella forma `route(Sender, Receiver, Prefix, Route)`, e viene usato per rappresentare l'annuncio di un percorso da un AS a un altro.

Questa distinzione consente di separare la conoscenza interna di un AS dagli instradamenti propagati nella relazione BGP.

Verifica della coerenza tra i percorsi

Per garantire che un cliente `As2` segua coerentemente i percorsi appresi dal proprio *provider* `As1`, è possibile definire una clausola di controllo di coerenza tra i rispettivi percorsi verso un prefisso IP `P`:

Implementazione in *Prolog*

```
% La configurazione è incoerente se As2 ha un percorso diverso
da quello appreso da As1
false :-
    provider(as1, as2),
    route(as1, P, Rp1),
    route(as2, P, Rp2),
    \+ (Rp2 = [as2 | Rp1]).
```

Questa clausola segnala un errore logico nel sistema. Essa fallisce (cioè rende il programma incoerente) quando si verifica quanto segue:

`provider(as1, as2)` stabilisce che esiste una relazione di tipo fornitore-cliente tra i due AS.

`As1` possiede un percorso `Rp1` verso un prefisso `P`:

```
route(as1, P, Rp1)
```

`As2` possiede un percorso `Rp2` verso lo stesso prefisso `P`:

```
route(as2, P, Rp2)
```

Ma il percorso di As2 non è coerente con quello di As1, ovvero:

```
\+ (Rp2 = [as2 | Rp1])
```

Quindi, As2 non ha semplicemente anteposto se stesso al percorso appreso da As1, come invece dovrebbe fare secondo la semantica della relazione fornitore-cliente. Questa clausola consente di rilevare configurazioni anomale o non conformi alle aspettative del modello BGP cliente-fornitore, garantendo che i percorsi appresi siano propagati in modo coerente e verificabile.

La relazione di transito tra due sistemi autonomi può essere rappresentata mediante le seguenti clausole *Prolog*, distinguendo tra fatti specifici e regole generali.

Fatti (istanze concrete)

```
% Il sistema autonomo 6762 è provider di 218765
provider(6762, 218765).
```

```
% Esiste una sessione BGP tra i due AS
neighbor(6762, 218765).
neighbor(218765, 6762).
```

Clausole generali

```
% AS1 ha una tabella T1 contenente instradamenti verso P
route(AS1, AS2, P, Rp1) :-
```

```
    route_in_table(AS1, T1, P),
    firt(P),
    member(Rp1, T1),
    next_hop(Rp1, AS2),
    prefix(Rp1, P).
```

```
% AS2 costruisce la propria tabella T2 in base a quanto
appreso da AS1
```

```
route(AS2, AS1, P, Rp2) :-
    route_in_table(AS2, T2, AS1),
    member(P, [firt]),
    member(Rp2, T2),
```

```

    next_hop(Rp2, P),
    prefix(Rp2, P).

```

Clausole ausiliarie

```

% Tabella di routing di AS1 per il prefisso P
route_in_table(AS1, T1, P) :-
    findall(R, route(AS1, _, P, R), T1).

```

```

% Tabella di routing di AS2 relativa a instradamenti ricevuti
da AS1
route_in_table(AS2, T2, AS1) :-
    findall(R, route(AS2, AS1, _, R), T2).

```

6.2.3 CLIENTE-FORNITORE

Il sistema autonomo AS2 (il cliente) annuncia i propri prefissi e quelli appartenenti a sua volta ai propri clienti. Per ciascun prefisso IP p annunciato dall'AS2 o dai suoi clienti (AS2_SET) esiste almeno un instradamento nella tabella di *routing* T2 di AS2 che conduce a quel prefisso; allo stesso modo esiste un instradamento nella tabella di *routing* T1 di AS1 che conduce ad AS2.

$$\forall p \in \text{AS2_SET} : \exists rp,2 \in T2 \wedge \exists rp,1 \in T1$$

```

% Insieme dei prefissi annunciati dal cliente As2
as2_set(as2, [p1, p2, p3]).

```

```

% As2 annuncia il suo AS_SET ad As1
% As1 accetta l'AS_SET
% As1 installa gli instradamenti nella tabella di routing
in_as_set(As2, P) => route(As2, P, Rp2), route(As1, P, Rp1),
Rp1 = [As1 | Rp2]

```

Questa relazione può essere rappresentata dalle seguenti clausole *Prolog*:

```

neighbor(as1, as2).
neighbor(as2, as1).

```

```

route(as2, as1, P, Rp1) :-
    route_in_table(as2, T2, P),
    as2_set(as2, Set),
    member(P, Set),
    member(Rp1, T2),
    next_hop(Rp1, as1),
    prefix(Rp1, P).

route(as1, as2, P, Rp2) :-
    route_in_table(as1, T1, as2),
    as2_set(as2, Set),
    member(P, Set),
    member(Rp2, T1),
    next_hop(Rp2, P),
    prefix(Rp2, P).

route_in_table(as2, T2, P) :-
    findall(R, route(as2, _, P, R), T2).

route_in_table(as1, T1, as2) :-
    findall(R, route(as1, as2, _, R), T1).

```

6.2.4 PEERING

Assumiamo che AS1 e AS2 siano due sistemi autonomi che hanno una sessione BGP. Per ciascun prefisso IP p annunciato da AS1 o dai suoi clienti (AS1_SET) esiste almeno un instradamento verso p nella tabella di *routing* di AS2.

Per ciascun prefisso IP q annunciato da AS2 o dai suoi clienti (AS2_SET) esiste almeno un instradamento verso q nella tabella di *routing* di AS1.

$$\forall p \in \text{AS1_SET} : \exists rp,2 \in T2$$

$$\forall q \in \text{AS2_SET} : \exists rq,1 \in T1$$

% As1 e As2 sono peer se e solo se

% As1 annuncia il suo AS_SET a As2 E As2 annuncia il suo AS_SET ad As1

Questa relazione può essere rappresentata dalle seguenti clausole *Prolog*:

```
neighbor(as1, as2).
```

```
neighbor(as2, as1).
```

```
as1_set(as1, [p1, p2, p3]).
```

```
as2_set(as2, [q1, q2, q3]).
```

```
route(as1, as2, P, Rp2) :-
```

```
    route_in_table(as1, T1, P),
```

```
    as1_set(as1, Set1),
```

```
    member(P, Set1),
```

```
    member(Rp2, T1),
```

```
    next_hop(Rp2, as2),
```

```
    prefix(Rp2, P).
```

```
route(as2, as1, Q, Rq1) :-
```

```
    route_in_table(as2, T2, Q),
```

```
    as2_set(as2, Set2),
```

```
    member(Q, Set2),
```

```
    member(Rq1, T2),
```

```
    next_hop(Rq1, as1),
```

```
    prefix(Rq1, Q).
```

```
route_in_table(as1, T1, P) :-
```

```
    findall(R, route(as1, as2, P, R), T1).
```

```
route_in_table(as2, T2, Q) :-
```

```
    findall(R, route(as2, as1, Q, R), T2).
```


7 CHIARIMENTI SUL METODO

Illustrati sin qui i rapporti che intercorrono tra i sistemi autonomi e i ruoli che ciascun AS può assumere, è necessario chiarire quali sono i motivi per i quali si escludono i rapporti di *peering* dall'esame che le azioni MANRS richiedono sulle politiche di instradamento e come vengono trattati alcuni sistemi autonomi particolari definiti come *Tier-1*.

7.1 TRANSITI COME CRITERIO DI ESISTENZA DI UN AS

Ai fini dell'esistenza amministrativa e dell'operatività su Internet di un ISP, i rapporti di transito (generalmente considerati un requisito per essere *multi-homed* cioè presenti su Internet grazie all'interconnessione a due diversi sistemi autonomi) rivestono un ruolo primario e indispensabile (definizione valida per tutti gli AS tranne che per i *tier-1* i quali, pur soddisfacendo il requisito di essere *multi-homed*, tuttavia non usufruiscono di transiti per raggiungere tutti gli AS attivi su Internet) laddove invece i rapporti di *peering* se da una parte rappresentano quasi sempre (si consideri il caso dei *peering* remoti trattati più avanti) un importante *plus* per l'efficienza della rete (riduzione drastica della latenza) e per le politiche di instradamento (mantenimento locale del traffico locale), dall'altra sono solitamente opzionali per l'esistenza di un AS [70]. Questo ragionamento conduce a eleggere quello sui transiti come controllo principe per la verifica dei sistemi autonomi nell'ambito delle azioni propedeutiche alla certificazione MANRS. I rapporti di transito sono, per così dire, esposti alla luce del sole cioè pubblicamente osservabili, seppure con alcune avvertenze.

Innanzitutto partiamo dalla definizione di transito esplicitando l'assunto che nessun AS nel mondo di Internet è interconnesso a tutti gli altri contemporaneamente e in modo diretto. Dunque si può dire che AS1 è transito di AS2 quando consente al traffico proveniente da AS2 e destinato ad altri AS (con i quali AS2 non è interconnesso) appunto di transitare per la sua infrastruttura. Ad esempio, dato che AS2 non ha interconnessione con AS3, allora per raggiungerlo transita per AS1.

Definizione di sessione BGP:

$BGP_session = (AS1, AS2, I1, I2, E1, E2, AS_PATH)$

Dove

I1 è l'insieme dei prefissi che AS1 può importare da AS2 con un certo *AS_PATH*.

I2 è l'insieme dei prefissi che AS2 può importare da AS1 con un certo *AS_PATH*.

E1 è l'insieme dei prefissi che AS1 può esportare verso AS2 con un certo AS_PATH.

E2 è l'insieme dei prefissi che AS2 può esportare verso AS1 con un certo AS_PATH.

AS_PATH è l'insieme degli Autonomous Systems che compongono il percorso del prefisso BGP scambiato tra AS1 e AS2.

Definizione di transito:

$$\text{Transit}(AS_x, AS_y) \Leftrightarrow \exists AS_z [AS_PATH(AS_y, AS_z) \wedge \neg \text{Interconnection}(AS_y, AS_z) \wedge AS_PATH(AS_x, AS_z)]$$

AS_x è transito di AS_y se esiste almeno un AS_z tale che l'AS_PATH da AS_y a AS_z è definito, non c'è interconnessione diretta tra AS_y e AS_z, e contemporaneamente l'AS_PATH da AS_x a AS_z è definito.

Questo è il punto di vista di AS2, ma quali sono invece le caratteristiche che AS1 deve avere per poter erogare transito? AS1 deve avere una tabella di *routing* in grado di instradare il traffico verso qualunque destinazione esistente (e attiva) su Internet, non importa quali siano i mezzi che AS1 ha impiegato per ottenerla (altri transiti, *peering* bilaterali, *peering* multilaterali, PNI ecc). Il caso più semplice è che AS1 abbia, nella propria tabella di *routing*, un instradamento di *default* (IPv4 0.0.0.0/0 e/o IPv6 ::/0) verso il proprio *upstream* (altro nome del *provider* che eroga transito). Tuttavia, quest'ultimo caso è una eccezione che non dovrebbe coinvolgere gli AS che forniscono (a titolo oneroso) un servizio di transito poiché, dato che i RIR richiedono che ciascun AS sia *multi-homed* (cioè che abbia almeno due *upstream provider* distinti), sarebbe assai limitata la possibilità di gestire politiche di *traffic engineering*.

Per questo motivo i rapporti di transito consistono nella trasmissione, attraverso il BGP, di tutti i *best-path* di cui l'AS erogante è in possesso. L'insieme degli AS che si interconnettono tra loro filtrando gli instradamenti di *default* si chiama *default-free zone* (DFZ). Ora, se AS1 è parte della DFZ (in qualità di transito), ha la possibilità di trasmettere ad AS2 sia un instradamento di *default* (probabile solo se AS2 sia un cosiddetto *stub* AS) sia la sua intera tabella di *routing* via BGP (stragrande maggior parte dei rapporti). Considerando questo secondo caso il più plausibile, si ha una relazione nella quale AS1 trasmette ad AS2 tutti i *best-path* che ha selezionato e, a sua volta, AS2 (che in questo esempio non ha altri AS come clienti) trasmette ad AS1 i prefissi originanti dal proprio AS (cioè quelli assegnati dai RIR).

Definizione di FIRT:

$\text{FIRT} = \{ p \mid \text{esiste AS attivo per cui } p \text{ è prefisso di AS} \}$

$P_{\text{firt}} = \{ P_{\text{AS}_n} \}$

Insieme dei prefissi di ogni AS attivo su Internet.

Definizione di Prefisso locale:

$P_{\text{local}} \rightarrow \text{AS_PATH } \{ \}$

Il prefisso che origina localmente a un AS ha un AS_PATH vuoto

Definizione di Transito:

$\text{Transit}(\text{AS}_1, \text{AS}_2) \Leftrightarrow \text{BGP}_{\text{session}} = (\text{AS}_1, \text{AS}_2, I_1, I_2, E_1, E_2, \text{AS_PATH})$

dove $I_1 = (P_{\text{local}} \cup P_{\text{cc}})$ $I_2 = (P_{\text{firt}})$, $E_1 = I_2$, $E_2 = I_1$

AS1 è transito di AS2

7.1.1 CUSTOMER-CONE

Nel caso in cui AS2 eroghi a sua volta un servizio di transito per altri AS (per esempio ad AS10, AS20 e AS30), allora dovrebbe trasmettere ad AS1 anche i prefissi dei propri clienti, il cosiddetto *Customer-cone*:

Nel campo semantico di RPSLng [\[110\]](#) (*Routing Policy Specification Language next generation*) è un linguaggio usato dagli operatori di rete per descrivere le politiche di instradamento) tale insieme di AS viene esplicitato attraverso la valorizzazione nell'oggetto *as-set*: del campo *members*: dove si inseriscono, oltre il proprio AS, i numeri di AS dei propri clienti, ad esempio:

`as-set: AS-IBSCUST`

`members: AS3269, AS2162, AS2164, AS2593, AS2594, AS2595,
AS3228, AS3573, AS3275, AS5382, AS5394, AS5397, AS5502,
AS6734, AS6760, AS112`

Definizione di Customer-cone:

$P_{\text{cc}} = \cup P_{\text{clients}}$

Unione dei prefissi di ciascun AS cliente (cioè a cui si dà transito)

Queste due ultime fattispecie determinano la formazione di AS_PATH (che sono una sequenza ordinata di numeri di AS) di lunghezza leggermente diversa.

$AS_PATH = \{AS_1, AS_2, \dots, AS_n\}$

$n \geq 0$

oppure

$AS_PATH = \{ \}$

nel caso di prefissi originati localmente

Infatti, dalla prospettiva di AS_1 , gli AS_PATH che deve considerare sarebbero:

$AS_1 \ AS_2$

$AS_1 \ AS_2 \ AS_{10}$

$AS_1 \ AS_2 \ AS_{20}$

$AS_1 \ AS_2 \ AS_{30}$

Da questo punto di vista l'individuazione dell'AS in funzione di transito è immediata perché emerge dall'ordine della sequenza dell' AS_PATH ; così è possibile osservare che AS_1 è transito per AS_2 , che AS_2 è transito per AS_{10} , AS_{20} e AS_{30} . Si tratta di una conclusione che può essere desunta facilmente ogni volta che si esamina la tabella BGP di un *route-collector*³ ma solo per AS_PATH più lunghi di due AS. Il motivo sta nel fatto che l'accordo con i sistemi autonomi dei *route-collector* prevede che l'afferente trasmetta l'intera tabella di *routing* senza filtri e che, a sua volta, il *collector* non annunci alcun prefisso [71]. A dire la verità questo punto è stato a lungo dibattuto proprio per il fatto che non ci fosse, a livello globale, una regola unica per gestire l'afferenza a un *route-collector*. Anche l'ultimo tentativo, del 2022, non è andato in porto [111].

$\forall AS_{RC}, AS_2. \ BGP_{session} = (AS_{RC}, AS_2, I_{RC}, I_2, E_{RC}, E_2, AS_PATH)$

dove $I_{RC} = (P_{firt})$, $I_2 = ()$, $E_{RC} = ()$, $E_2 = (P_{firt})$

Per ogni coppia di AS, di cui uno sia di un route-collector, si instaura una sessione BGP all'interno della quale vengono scambiati AS_PATH in modo tale che il route-collector importi tutta la FIRT esportata da AS_2 . A sua volta AS_2 non importa nulla, né il route-collector esporta AS_PATH .

Questa circostanza fa sì che risulti praticamente impossibile, dall'esame di una sola tabella BGP, determinare se il rapporto tra gli unici due AS che compongono l' AS_PATH sia di transito o di *peering* (tranne nel caso dei *tier-1* spiegato più avanti) poiché in entrambi i casi i

³ Il *route-collector* è una macchina, solitamente gestita da enti di ricerca non direttamente coinvolti con l'industria di Internet, che colleziona dati di instradamento provenienti da molteplici sistemi autonomi da tutto il mondo che volontariamente aderiscono a stabilire sessioni BGP per trasmettere le proprie tabelle di *routing*.

prefissi comparirebbero presso il *route-collector* con il medesimo AS_PATH. Ecco un esempio di AS_PATH corto (per un prefisso esaminato appunto dalla prospettiva del *collector*) dove AS100 è afferente al *route-collector* e contemporaneamente sia transito per AS1:

AS100 AS1

Di seguito invece un esempio di AS_PATH dove AS100 è afferente al *route-collector* e contemporaneamente abbia un *peering* con AS1:

AS100 AS1

Si capisce a questo punto come basti che l'AS_PATH, dal punto di vista del *collector*, sia costituito di almeno tre AS per essere certi che tra l'ultimo AS a destra e il penultimo ci sia un rapporto di transito (cioè dove, procedendo da sinistra verso destra, il penultimo faccia da transito all'ultimo).

$\forall AS_x, AS_y \in AS_PATH$ se $\neg Tier1(AS_x) \wedge \neg Tier1(AS_y) \wedge AS_x \neq AS_y \Rightarrow$
 $Transito(AS_x, AS_y)$

Per ogni coppia adiacente di AS (AS_x, AS_y) nell'AS_PATH, se entrambi gli AS non sono tier-1 e sono distinti (AS_x≠AS_y), allora AS_x fornisce transito a AS_y (cioè l'AS a sinistra dà transito a quello a destra)

AS_x, AS_y possono essere non distinti per AS_PATH allungati artificialmente (prepending)

7.2 L'ESCLUSIONE DEI *PEERING*

I *peering*, al contrario, non sono facilmente desumibili dall'osservazione di una tabella BGP per ragioni inerenti la natura stessa del rapporto. Alcuni studiosi hanno tentato di reperire un metodo per far emergere i rapporti di *peering* attraverso la collezione di migliaia di tabelle BGP importate da altrettanti punti di osservazione privilegiati come i *route-collector*. Metodi che, in alcuni casi, hanno dato vita a strumenti disponibili *on-line* che, purtroppo, si sono rivelati assai imprecisi e incapaci di produrre risultati completi e affidabili. Altri ci hanno provato combinando le tabelle BGP importate dai *route-collector* con le dichiarazioni che gli AS hanno rilasciato via RPSLng agli IRR o a soggetti aggregatori terzi come peeringDB (<https://www.peeringdb.com/>), dati combinati anche alle matrici di *peering* esposte pubblicamente da alcuni IXP nel mondo. Anche in questo secondo caso i risultati sono tutt'altro che esaustivi e accurati.

Si considerano *peering* delle sessioni BGP stabilite tra pari, cioè tra AS che più o meno si equivalgono o tra AS, anche molto diversi tra loro per dimensioni, che però hanno valide

motivazioni per ritenere vantaggioso il collegamento. Tradizionalmente le sessioni BGP paritarie si configurano a titolo gratuito proprio per l'esistenza di importanti interessi reciproci tra due AS. A volte tuttavia sul mercato si osservano dei particolari frangenti nei quali alcuni AS sono molto corteggiati per rapporti di *peering* e che, di conseguenza, concedono la configurazione solo a seguito della sottoscrizione di un contratto a titolo oneroso.

Può essere il caso di introdurre qui l'utilità delle PNI (*Private Network Interconnection*) nell'ambito delle *facility* messe a disposizione dagli IXP o da altre entità dove sono presenti le infrastrutture facenti capo a molteplici AS. Mentre la maggiore parte delle interconnessioni tra AS avviene sull'*hardware* offerto dai punti di interscambio (chiamato *IP Fabric*), alcuni altri rapporti possono rimanere privati e possono essere stabiliti sulla base di accordi che esulano dalle regole imposte dagli IXP. Nascono così delle interconnessioni, chiamate PNI, tra due soggetti che si accordano per scambiare traffico in modo riservato cioè senza fare uso dell'infrastruttura pubblica condivisa, decidendo autonomamente le regole del gioco (valide solo tra di loro naturalmente): *peering* a pagamento, transito, trasporto ecc.

Dunque la particolarità citata nel testo, cioè quella di un *peering* a titolo oneroso con un AS, ritenuto molto interessante, che non partecipa all'infrastruttura pubblica (e gratuita), trova appunto casa nelle PNI.

La maggior parte dei rapporti di *peering* vede la luce nei NAP, *Network Access Point* (chiamati anche IXP, *Internet Exchange Point*), cioè in infrastrutture appositamente dedicate all'incontro tra AS diversi (tipicamente gli AS *tier-1* non afferiscono agli IXP pubblici o, quanto meno, non si presentano sulla LAN di *peering*).

Si tratta di una condizione logica più che fisica. Infatti l'afferenza agli IXP può essere materiale (cioè con apparati e circuiti presenti nelle *facility* dell'IXP) o virtuale (cioè senza apparati o circuiti di proprietà *in loco*, ma attraverso trasporto effettuato da altri afferenti) e così, conseguentemente, i rapporti di *peering* possono essere locali o remoti (con eventuali ripercussioni sulle prestazioni della rete).

Tradizionalmente le sessioni si stabiliscono uno a uno attraverso accordi che si prendono in modo più o meno informale: una modalità che, soprattutto con l'affollarsi dei punti di interscambio, ha mostrato la grave incapacità di scalare. Perciò è stata introdotta una nuova tecnica per la realizzazione dei *peering*, stavolta non più in modo bilaterale, ma in modo multilaterale. In altre parole gli IXP oggi, nella lista di molteplici servizi per gli operatori di rete, mettono a disposizione anche un apparato chiamato *Route Server* [72] che ha la caratteristica di poter intrattenere rapporti BGP con tutti gli afferenti e di riflettere gli

instradamenti di ciascuno a tutti gli altri (senza modificare l'AS_PATH). In tal modo a ogni AS è sufficiente instaurare una sessione BGP con il *Route Server* per ricevere gli stessi instradamenti che gli sarebbero stati trasmessi su N tradizionali sessioni di *peering*. Se il numero degli afferenti a un IXP è N, le configurazioni che ciascun AS deve mettere in piedi per stabilire un rapporto di *peering* con tutti gli altri è di N-1. Invece, in presenza di un *route-server* (o più d'uno per ridondanza) è sufficiente configurare una sessione di *peering* con il *route-server* per ricevere automaticamente gli instradamenti di tutti gli altri AS che con quel *route-server* hanno stabilito un rapporto di *peering*.

La sessione di *peering* si differenzia da quella di transito anche per la natura degli annunci che si esportano e che si importano. Mentre nel transito si accetta tutta la tabella degli instradamenti dall'*upstream provider* e gli si trasmettono solo i propri prefissi o i propri prefissi e quelli dei propri eventuali clienti, nel rapporto di *peering* ciascuno trasmette e accetta solo i prefissi propri dell'AS e quelli del cosiddetto *Customer-cone*. Dunque è un rapporto che ha rilevanza solo tra le parti che partecipano e che, per questo motivo, non si caratterizza negli AS_PATH osservabili dal punto di vista dei *route-collector*.

Definizione di *peering*:

$$\text{Peering}(AS_1, AS_2) \Leftrightarrow \text{BGP}_{\text{session}} = (AS_1, AS_2, I_1, I_2, E_1, E_2)$$

dove $I_1 = (P_{\text{local}} \cup P_{\text{cc}})$ $I_2 = (P_{\text{local}} \cup P_{\text{cc}})$, $E_1 = I_2$, $E_2 = I_1$

AS1 ha una relazione di peering con AS2 quando, all'interno della sessione BGP che stabiliscono, ciascuno importa solo i prefissi dell'altro e del Customer-cone dell'altro, e contemporaneamente, ciascuno esporta solo i propri prefissi e quelli del proprio Customer-cone

7.2.1 AS SPECIALI: I TIER-1

In questo contesto è necessario riferire di alcuni speciali AS che non attingono ad altri AS per il transito: si chiamano infatti *Transit-Free Network* perché intrattengono esclusivamente rapporti di *peering* (*settlement-free peering*) tra loro. Sono sedici AS chiamati di livello 1, *tier-1*, che dunque stabiliscono tra loro rapporti di *peering*, non acquisiscono transito (semplicemente perché non ne hanno bisogno) e vendono transito (solitamente all'ingrosso) a tutti gli altri. Un numero che può leggermente variare nel tempo a seconda delle mutazioni dei rapporti tra gli operatori di rete. In questo caso consideriamo solo i *tier-1* su scala

mondiale (sia IPv4 sia IPv6) e non quelli rilevanti a livello continentale o quelli che sono *transit-free network* solo in IPv4 o solo in IPv6:

- 1 AS701 Verizon
- 2 AS1239 T-Mobile
- 3 AS1299 Arelion
- 4 AS2914 NTT
- 5 AS3257 GTT
- 6 AS3320 DTAG
- 7 AS3356 Lumen
- 8 AS3491 PCCW
- 9 AS5511 Orange
- 10 AS6453 Tata
- 11 AS6461 Zayo
- 12 AS6762 Sparkle
- 13 AS6830 Liberty Global
- 14 AS7018 AT&T
- 15 AS7922 Comcast
- 16 AS12956 Telxius

$AS_{tier1} = \{AS701, AS1239, AS1299, AS2914, AS3257, AS3320, AS3356, AS3491, AS5511, AS6453, AS6461, AS6762, AS6830, AS7018, AS7922, AS12956\}$

Quindi gli AS non appartenenti alla cerchia dei *tier-1* sono per definizione clienti di un *tier-1* o, al massimo, clienti di cliente (e così via) di un *tier-1*. Va specificato che questa particolare proprietà dei *tier-1* va declinata sia per IPv4 sia per IPv6, di conseguenza si osserva che l'insieme degli AS *tier-1* del mondo IPv4 non coincida con quello del mondo IPv6. Emblematico è il caso di AS6939, Hurricane Electric, che è *tier-1* per IPv6 ma non per IPv4; oppure il caso di Cogent, AS174, che è stato *tier-1* per IPv4 ma non per IPv6.

Quando dunque si rinviene il numero di un AS *tier-1* in un AS_PATH, si può pacificamente dedurre che sia transit dell'AS posizionato immediatamente alla sua destra (tranne nel caso di altro AS *tier-1*, caso in cui il rapporto è di *peering*), ovunque la coppia si posizioni nella sequenza. Oltre questo si può dedurre che, se alla sua sinistra si trova un AS diverso da *tier-1*, faccia da transit anche a quest'ultimo. Se invece alla sua sinistra c'è un altro *tier-1*, allora si tratta di un rapporto di *peering*, come precedentemente menzionato.

Ecco un esempio chiarificatore:

AS6762 è un *tier-1*

AS100 è afferente al *route-collector*

desideriamo osservare l'AS_PATH per il prefisso 1.2.3.0/24 che origina da AS20

1.2.3.0/24 AS100 AS6762 AS1 AS2 AS20

Per tutto quanto esposto finora, l'osservazione di questa sequenza fa scaturire la dichiarazione di alcuni fatti:

AS2 è transito di AS20

AS1 è transito di AS2

AS6762 è transito di AS1

AS6762 è transito di AS100

$\forall AS_1, AS_{tier1} \in AS_PATH \Rightarrow Transit(AS_{tier1}, AS_1)$

Per ogni coppia di AS distinti ($AS_{tier1} \neq AS_1$) in un AS_PATH, se AS_{tier1} è tier-1 e AS_1 non è tier-1, allora AS_{tier1} fornisce transito ad AS_1

Altro esempio:

AS6762 è un *tier-1*

AS6762 è afferente al *route-collector*

desideriamo osservare l'AS_PATH per il prefisso 1.2.3.0/24 che origina da AS20

1.2.3.0/24 AS6762 AS1 AS2 AS20

Osservazioni:

AS2 è transito di AS20

AS1 è transito di AS2

AS6762 è transito di AS1

Altro esempio:

AS6762 è un *tier-1*

AS 2914 è un *tier-1*

AS100 è afferente al *route-collector*

desideriamo osservare l'AS_PATH per il prefisso 1.2.3.0/24 che origina da AS20

1.2.3.0/24 AS100 AS6762 AS2914 AS1 AS2 AS20

$\forall AS_x, AS_y \text{ se } AS_x \neq AS_y \wedge Tier1(AS_x) \wedge Tier1(AS_y) \Rightarrow$

$Peering(AS_x, AS_y)$

se entrambi gli AS distinti sono tier-1, allora il loro rapporto è di peering

Osservazioni:

AS2 è transito di AS20

AS1 è transito di AS2

AS2914 è transito di AS1

AS6762 è *peer* di AS2914

AS6762 è transito di AS100

Infine:

AS6762 è un tier-1

AS 2914 è un tier-1

AS100 è afferente al *route-collector*

AS6762 è afferente al *route-collector*

AS200 è afferente al *route-collector*

desideriamo osservare l'AS_PATH per il prefisso 1.2.3.0/24 che origina da AS20

1.2.3.0/24	AS100 AS6762 AS2914 AS1 AS2 AS20
1.2.3.0/24	AS100 AS6762 AS1 AS2 AS20
1.2.3.0/24	AS6762 AS1 AS2 AS20
1.2.3.0/24	AS100 AS200 AS20
1.2.3.0/24	AS200 AS20

Osservazioni:

AS2 è transito di AS20

AS1 è transito di AS2

AS2914 è transito di AS1

AS6762 è *peer* di AS2914

AS6762 è transito di AS100

AS6762 è transito di AS1

AS200 è transito di AS20

AS100 è *peer* o transito di AS200

7.3 CONCLUSIONI

Da questi chiarimenti sul metodo adottato per la conformità delle politiche di instradamento dei sistemi autonomi alle azioni MANRS, emerge chiaramente quanto differiscono i concetti di transito e *peering*. Il transito emerge come criterio fondamentale, poiché permette la trasmissione del traffico verso destinazioni non direttamente connesse, garantendo la funzionalità amministrativa e operativa degli AS. Questo aspetto è particolarmente rilevante per la certificazione MANRS poiché la pubblica osservabilità dei rapporti di transito, attraverso la trasmissione dei *best-path* via BGP, facilita il monitoraggio e la verifica delle politiche di instradamento. D'altronde l'osservazione dei rapporti di *peering* rimane una sfida ancora aperta e complessa data la loro natura paritaria non dichiarata nelle politiche di instradamento. Per questo tali rapporti non vengono considerati nel percorso di verifica di conformità MANRS.

8 RoSe-T

Routing Security Tool ha l'obiettivo di implementare un metodo in grado di verificare le configurazioni dei dispositivi in uso all'interno dei sistemi autonomi sfruttando i dati estratti da fonti autorevoli e affidabili (*Regional Internet Registry* e *route-collector*). In generale, lo strumento consente di garantire che una determinata configurazione di *router* sia conforme alle linee guida che MANRS prevede per gli operatori di rete. In particolare esegue il controllo per convalidare le azioni già dettagliatamente illustrate nel precedente capitolo:

azione 1: prevenire la propagazione di informazioni di *routing* non corrette;

azione 2: impedire l'ingresso o l'uscita dalla rete di pacchetti con indirizzo IP sorgente falsificato;

azione 4: verificare che le politiche di instradamento che gli operatori di rete documentano pubblicamente negli IRR trovino una conferma nell'osservazione delle tabelle BGP apprese dai *route-collector*.



L'azione 3 MANRS invece non può essere convalidata automaticamente poiché implica la verifica delle informazioni di contatto del sistema autonomo. Si tratta di una attività di *auditing* condotta manualmente anche da RIPE NCC attraverso l'*Assisted Registry Check* (ARC) con l'obiettivo di fornire un supporto personalizzato al titolare di risorse (membro di RIPE NCC) per aiutarlo a mantenere i suoi dati nel registro RIPE corretti e accurati. Un ARC può essere avviato su richiesta di un membro del RIPE NCC o a seguito di estrazione da parte del RIPE NCC o per dirimere una specifica questione inerente un membro.

Prima di giungere alla versione attuale di RoSe-T sono stati creati dei prototipi che si sono rivelati utilissimi per alcune riflessioni architetturali. La prima divulgazione di ROSE-T, con

un altro nome (RS4LK [96]), è stata fatta durante un *workshop* ospitato da “FT Wholesale Tour” il 19 aprile 2023 a Pisa, Italia. Un primo abbozzo del lavoro che ha riscosso l’interesse degli operatori di rete presenti all’evento. Il progetto prevedeva l’analisi della configurazione dei *router* cosa che per essere realizzata necessitava di strumenti in grado di comprendere autonomamente il *vendor* del dispositivo, i transiti e i clienti. A questo scopo sono stati valutati alcuni software esistenti come Batfish, Bagpipe e BGPVerif [98]. Tuttavia generalmente si tratta di ricerche abbandonate allo stato embrionale che mancavano di alcuni requisiti (non contemplano infatti alcune specifiche caratteristiche del BGP, oppure non hanno supporto per IPv6); inoltre i *vendor* sono molteplici e ciascuno impiega una sintassi (o più di una) che evolve nel tempo, di conseguenza quegli strumenti non si presentavano adatti allo scopo.

Quindi all’epoca, per l’applicazione del metodo studiato, è stata apportata una estensione a *Batfish* così che il *software* potesse essere valido per un semplice *Proof of concept*. Subito dopo, con la necessità di integrare più *vendor* diversi, è stato abbandonato *Batfish* per *Antlr* [97], un potente generatore di *parser* per la lettura, l’elaborazione, l’esecuzione o la traduzione di testo strutturato. *Antlr* si è dimostrato un ottimo strumento ma, in un certo senso, sovradimensionato rispetto alla necessità di RoSe-T. E dunque, successivamente è stato implementato un *parser* originale capace di comprendere la sintassi di alcuni *vendor*. Tutto ciò ha portato all’individuazione del *software kathará* [75] che è “un sistema di emulazione di rete *open source* basato su *container* utile per dimostrazioni/lezioni interattive, per testare reti di produzione realizzate in un ambiente *sandbox* o per sviluppare nuovi protocolli di rete”. In questo contesto, serve per emulare uno scenario di rete virtuale in cui il *router* (del candidato MANRS) interagisce realisticamente con fornitori e clienti. Il *modus operandi* può essere schematizzato in cinque fasi.

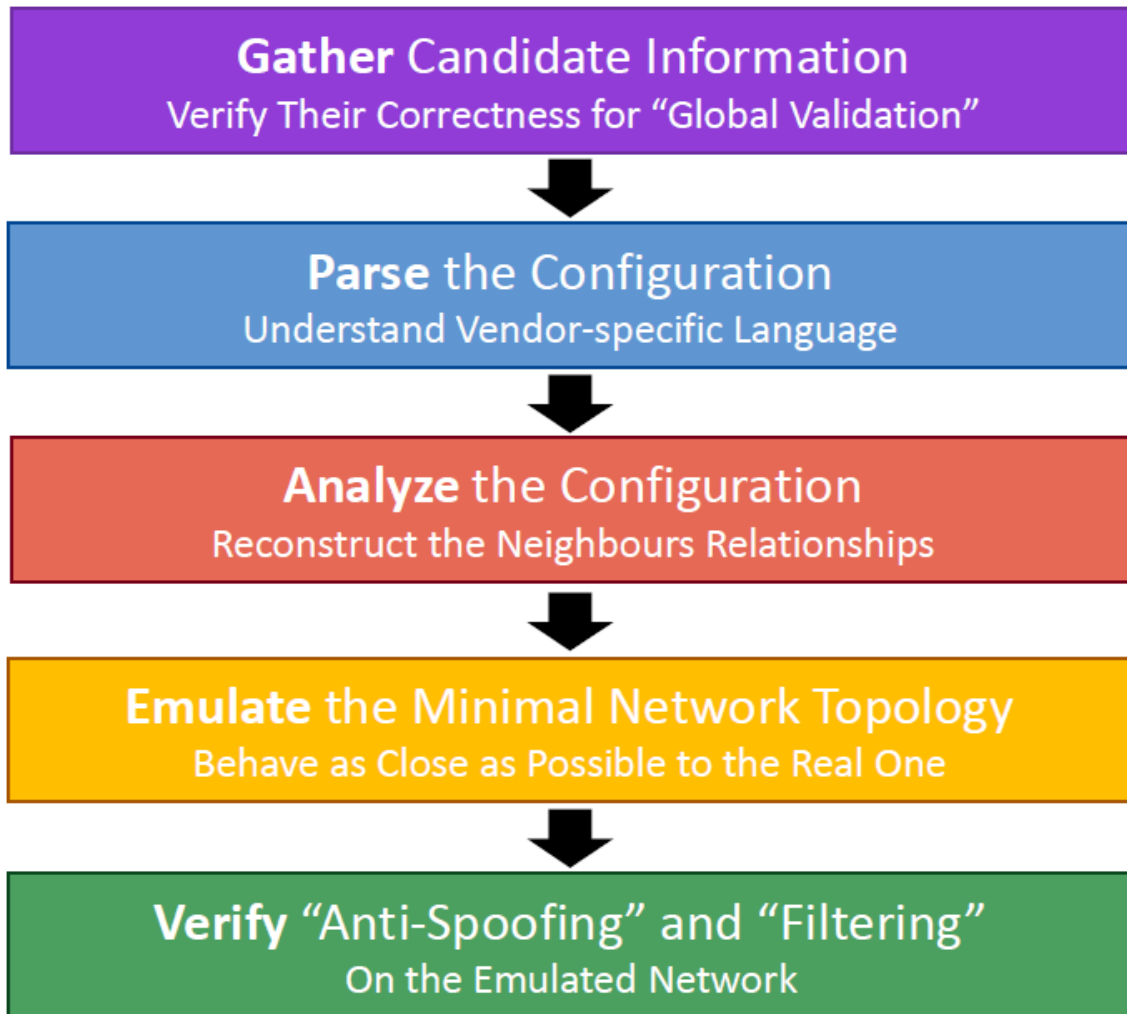


Fig. 3: Le cinque fasi di RoSe-T

8.1 LE CINQUE FASI OPERATIVE

8.1.1 FASE 1: RACCOLTA E VERIFICA DELLE INFORMAZIONI SUL CANDIDATO

In questa fase il sistema controlla le informazioni globali del candidato allo scopo di convalidare le politiche di instradamento pubblicate. Dunque RoSe-T verifica che:

1. le reti annunciate al transito siano presenti negli IRR;
2. le reti dichiarate negli IRR siano annunciate ai transiti.

8.1.2 FASE 2: PARSING DELLA CONFIGURAZIONE

In questa fase, RoSe-T analizza le informazioni richieste dalla configurazione del fornitore (utilizzando un *parser* personalizzato, basato su Batfish). Principalmente, estrae le informazioni sulle interfacce (nomi e indirizzi IP) e le sessioni BGP.

8.1.3 FASE 3: ANALISI DELLA CONFIGURAZIONE

RoSe-T analizza la configurazione per ricostruire le relazioni BGP con i dirimpettai. Integra le informazioni estratte dagli IRR e da un *dump* RIB attinto da un *route-collector* per dedurre la topologia e comprendere le relazioni.

8.1.4 FASE 4: EMULAZIONE DELLA TOPOLOGIA MINIMA DELLA RETE

Poi il sistema utilizza le informazioni elaborate per costruire una topologia di rete minima da emulare. Per costruire l'emulazione, RoSe-T fa uso di *kathará*. Il *router* candidato utilizzerà la configurazione originale secondo il sistema operativo del *vendor* usato, mentre gli altri sistemi autonomi sono emulati ciascuno come un *router* che esegue *FRRouting* (una *suite* di protocolli di *routing* Internet libera e a sorgente aperta per piattaforme Linux e Unix. Implementa BGP, OSPF, RIP, IS-IS, PIM, LDP, BFD, Babel, PBR, OpenFabric e VRRP, con supporto alfa per EIGRP e NHRP). [[108](#)]

8.1.5 FASE 5: VERIFICA DELLA CONFORMITÀ ALLE AZIONI MANRS

In questa fase il sistema sfrutta l'ambiente emulato per verificare le azioni MANRS 1 e 2.

Filtraggio (Azione 1): “Garantire la correttezza dei propri annunci e di quelli dei clienti verso le reti adiacenti”.

Per ogni cliente:

1. selezionare una sottorete non sovrapposta e annunciarla al *router* candidato;
2. attendere la convergenza del BGP;

3. controllare gli instradamenti ricevuti dal *provider* utilizzando il piano di controllo FRR.

***Anti-Spoofing* (Azione 2): “Abilitare la convalida dell'indirizzo sorgente almeno per le reti di clienti *stub single-homed*, per i propri utenti finali e per l'infrastruttura”.**

Per ogni *provider*:

1. il sistema crea un client nel sistema autonomo del *provider*;
2. assegna gli IP (v4/v6) a ciascun *client* creato;
3. invia pacchetti ICMP di tipo *spoofed*;
4. controlla se il pacchetto di tipo *spoofed* lascia l'AS del candidato.

8.2 DETTAGLIO DELLA FASE 1: RACCOLTA E VERIFICA DELLE INFORMAZIONI

Per prima cosa è necessario osservare le politiche di instradamento dichiarate da un sistema autonomo e pubblicate nel *database* di un IRR. Si consideri ad esempio la seguente politica di instradamento dichiarata (attraverso RPSLng) dal sistema autonomo AS59715 nel *database* di RIPE:

```
mp-import: afi ipv4.unicast from AS12874 accept any
mp-import: afi ipv6.unicast from AS12874 accept any
mp-import: afi ipv4.unicast from AS6762 accept any
mp-import: afi ipv6.unicast from AS6762 accept any
mp-import: afi ipv4.unicast from AS112 accept AS112
mp-import: afi ipv6.unicast from AS112 accept AS112
mp-export: afi ipv4.unicast to AS12874 announce AS-SBTAP
mp-export: afi ipv6.unicast to AS12874 announce AS-SBTAP
mp-export: afi ipv4.unicast to AS6762 announce AS-SBTAP
mp-export: afi ipv6.unicast to AS6762 announce AS-SBTAP
mp-export: afi ipv4.unicast to AS112 announce any
mp-export: afi ipv6.unicast to AS112 announce any
```

Si leggono due famiglie di indirizzamento, IPv4 e IPv6; per ciascuna, nella direzione *import*, vengono dichiarati due distinti fornitori (cioè AS con relazione di transito): AS6762 e AS12874 dai quali AS59715 accetta qualsiasi annuncio operino sulla sessione BGP instaurata (*accept any*). Nella direzione opposta, *export*, AS59715 annuncia ai due fornitori solo il proprio AS-SET (cioè i propri prefissi e quelli dei propri clienti, se presenti) definito tramite AS-SBTAP. Si nota anche una terza sessione BGP, quella con AS112 dal quale AS59715 accetta solo i prefissi di AS112 e al quale annuncia qualunque instradamento conosca. Da ciò si può facilmente dedurre che AS59715 eroga transito ad AS112 per entrambe le famiglie di indirizzamento. Dal *database* di RIPE è possibile estrarre i prefissi che hanno origine da AS59715:

```
185.5.200.0/22
185.5.200.0/23
185.5.202.0/24
185.5.203.0/24
2a02:cdc0::/29
2a02:cdc6:3074::/48
2a02:cdc6:3100::/48
```

Simile estrazione è possibile operare per AS112:

```
192.175.48.0/24
2620:4f:8000::/48
```

Da un rapido esame dell'oggetto AS-SET AS-SBTAP sul *database* di RIPE NCC [\[109\]](#) si evince che consiste solo di due membri, AS59715 e AS112, dunque non è necessario proseguire la ricerca di ulteriori oggetti nel *database*. Ciò che però è importante a questo punto è la verifica che quanto sia stato dichiarato negli IRR corrisponda a verità, cioè sia presente nell'ecosistema di Internet. Per tale attività si seleziona un punto di osservazione privilegiato come quello del “*RouteViews BGP Route Viewer project*” nato e mantenuto nell'Università dell'Oregon, USA. Dalla tabella BGP visibile presso uno dei *router* partecipanti al progetto si possono estrarre tutte le informazioni relative ai prefissi originanti nell'AS59715 per la famiglia di indirizzamento IPv4.

```
Network 185.5.200.0/23
Path 6939 6762 59715
```

Path 8283 6762 59715 i
Path 3267 1299 6762 59715 i
Path 3333 6762 59715 i
Path 1351 6939 6762 59715 i
Path 7018 6762 59715 i
Path 3356 6762 59715 i
Path 20912 3257 6762 59715 i
Path 19214 3257 6762 59715 i
Path 3549 3356 6762 59715 i
Path 2497 2914 6762 59715 i
Path 3303 6762 59715 i
Path 4901 6079 3257 6762 59715 i
Path 101 3356 6762 59715 i
Path 49788 1299 6762 59715 i
Path 852 3356 6762 59715 i
Path 3561 209 3356 6762 59715 i
Path 57866 6830 6762 59715 i
Path 20130 6939 6762 59715 i
Path 3257 6762 59715 i
Path 6939 6762 59715 i

Network 185.5.200.0/22

Path 8283 6762 59715 i
Path 3267 1299 12874 59715 i
Path 3333 6762 59715 i
Path 7018 6762 59715 i
Path 20912 12874 59715 i
Path 49788 12552 3303 12874 59715
Path 3549 3356 12874 59715 i
Path 19214 174 6762 59715 i
Path 6939 6762 59715 i
Path 20130 6939 6762 59715 i
Path 3356 12874 59715 i
Path 852 3303 12874 59715 i

Path 57866 6830 6762 59715 i
Path 101 11164 3303 12874 59715 i
Path 1351 6939 6762 59715 i
Path 4901 6079 3303 12874 59715 i
Path 3257 3356 12874 59715 i
Path 3561 3910 3356 12874 59715 i
Path 2497 3356 12874 59715 i
Path 3303 12874 59715 i

Network 185.5.202.0/24

Path 8283 1299 12874 59715 i
Path 3267 1299 12874 59715 i
Path 3333 3303 12874 59715 i
Path 7018 3356 12874 59715 i
Path 20912 12874 59715 i
Path 49788 12552 3303 12874 59715
Path 3549 3356 12874 59715 i
Path 6939 12874 59715 i
Path 20130 6939 12874 59715 i
Path 3356 12874 59715 i
Path 852 3303 12874 59715 i
Path 57866 3356 12874 59715 i
Path 1351 6939 12874 59715 i
Path 101 11164 3303 12874 59715 i
Path 4901 6079 3303 12874 59715 i
Path 3257 3356 12874 59715 i
Path 3561 3910 3356 12874 59715 i
Path 2497 3356 12874 59715 i
Path 3303 12874 59715 i

Network 185.5.203.0/24

Path 8283 6762 59715 i
Path 3267 1299 6762 59715 i
Path 3333 6762 59715 i

```
Path 1351 6939 6762 59715 i
Path 7018 6762 59715 i
Path 3356 6762 59715 i
Path 20912 3257 6762 59715 i
Path 3549 3356 6762 59715 i
Path 2497 2914 6762 59715 i
Path 3303 6762 59715 i
Path 4901 6079 3257 6762 59715 i
Path 101 3356 6762 59715 i
Path 49788 1299 6762 59715 i
Path 852 3356 6762 59715 i
Path 3561 209 3356 6762 59715 i
Path 57866 6830 6762 59715 i
Path 20130 6939 6762 59715 i
Path 3257 6762 59715 i
Path 6939 6762 59715 i
```

Se le dichiarazioni pubblicate da AS59715 sul *database* del RIPE sono veritiere, allora anche dal punto di vista del *router* afferente al progetto *RouteViews* si dovrebbe osservare che AS59715 ha solo due fornitori (transiti) e specificatamente AS6762 e AS12874 poiché deve sempre esserci una corrispondenza tra quanto si dichiara nelle politiche di instradamento e come effettivamente si comporta l'AS su Internet.

Gli archivi di *RouteViews* vengono popolati ogni due ore con *snapshot* della tabella BGP di un *route-collector*, dunque, per le operazioni di verifica sull'AS candidato, si usa il *file* più recente disponibile, ad esempio `rib.20231009.1400.bz2`. Una volta scaricato - al momento gli *snapshot* sono di circa 100 MB ciascuno - viene elaborato con uno strumento capace di estrarre velocemente dati sugli AS_PATH di interesse per il candidato. Lo strumento è *bgpgrep* afferente alla Micro BGP Suite [73] programmata dallo studioso italiano Lorenzo Cogotti [74]. Ecco un caso d'uso che mette sotto esame il candidato AS59715:

```
./bgpgrep ./rib.20231009.1400.bz2 -aspath "59715$" > 59715.bgp
```

Il comando estrae tutti gli AS_PATH, che terminano per 59715, presenti nello *snapshot* di uno dei *collector* eseguito il 9 ottobre 2023 alle ore 14:00.

Di seguito l'*output* del comando (alcuni dettagli irrilevanti sono stati omessi):

```
=|185.5.200.0/22|37100                                3303                                12874
59715|105.16.0.247|i|||NO_EXPORT|105.16.0.247
37100|1696730621|1
=|185.5.200.0/22|3549                                3356                                12874
59715|208.51.134.255|i|||3356:2    3356:22    3356:100    3356:123
3356:505    3356:901    3356:2074    3549:2018    3549:30840    12874:700
12874:3454|208.51.134.255    3549|1696459515|1
[...]
=|185.5.200.0/23|37100                                6762
59715|105.16.0.247|i|||NO_EXPORT|105.16.0.247
37100|1696730618|1
=|185.5.200.0/23|2914    6762    59715|129.250.1.71|i|||2914:420
2914:1005    2914:2000    2914:3000    6762:1    6762:92
6762:13980|129.250.1.71    2914|1696458965|1
[...]
=|185.5.202.0/24|37100                                3303                                12874
59715|105.16.0.247|i|||NO_EXPORT|105.16.0.247
37100|1696730621|1
=|185.5.202.0/24|6939                                12874
59715|64.71.137.241|i|||64.71.137.241    6939|1696545433|1
[...]
=|185.5.203.0/24|37100                                6762
59715|105.16.0.247|i|||NO_EXPORT|105.16.0.247
37100|1696730618|1
=|185.5.203.0/24|2914    6762    59715|129.250.1.71|i|||2914:420
2914:1005    2914:2000    2914:3000    6762:1    6762:92
6762:13980|129.250.1.71    2914|1696458965|1
```

Ogni riga che inizia con il segno “=” ha valori compresi tra “|”. I dati importanti per lo scopo di questa ricerca sono (guardando l’ultima riga): la rete (185.5.203.0/24), l’AS_PATH (2914 6762 59715) e il *neighbor* (129.250.1.71).

Parallelamente è necessario richiamare (come fatto in precedenza) quanto dichiarato dall’AS candidato all’interno del *database* del RIR di competenza territoriale e, nel dettaglio, è

necessario identificare quale sia la politica di instradamento resa pubblica. L'oggetto da considerare è *aut-num*: poiché contiene ciò che è di interesse alla verifica:

```
aut-num: AS59715
as-name: SBTAP-AS
org: ORG-CdSB1-RIPE
mp-import: afi ipv4.unicast from AS12874 accept any
mp-import: afi ipv6.unicast from AS12874 accept any
mp-import: afi ipv4.unicast from AS6762 accept any
mp-import: afi ipv6.unicast from AS6762 accept any
mp-import: afi ipv4.unicast from AS112 accept AS112
mp-import: afi ipv6.unicast from AS112 accept AS112
mp-export: afi ipv4.unicast to AS12874 announce AS-SBTAP
mp-export: afi ipv6.unicast to AS12874 announce AS-SBTAP
mp-export: afi ipv4.unicast to AS6762 announce AS-SBTAP
mp-export: afi ipv6.unicast to AS6762 announce AS-SBTAP
mp-export: afi ipv4.unicast to AS112 announce any
mp-export: afi ipv6.unicast to AS112 announce any
admin-c: SBT21-RIPE
tech-c: SBT20-RIPE
tech-c: AP7729-RIPE
status: ASSIGNED
mnt-by: RIPE-NCC-END-MNT
mnt-by: SBTAP-MNT
created: 2012-10-04T12:53:46Z
last-modified: 2022-10-24T14:18:26Z
source: RIPE
```

Ciò che va verificato è che da una parte i rapporti di transito espressi nella politica di instradamento pubblicata nel *database* del RIR trovino riscontro nella tabella BGP del *collector* e, dall'altra, che, sempre per le relazioni di transito, gli AS_PATH della tabella BGP corrispondano a quanto dichiarato nel *database* del RIR. Per condurre tale controllo è stato usato il linguaggio *Prolog* utile a trattare i dati come fatti e a costruire delle relazioni tra i fatti così da poter verificare alcune proprietà. Il programma dunque accetta come *input* due *file*: uno contenente gli AS_PATH della tabella BGP relativi all'AS candidato (59715.bgp), l'altro

contenente la politica di instradamento dichiarata dal candidato nel *database* del RIR (59715.rir).

```
./verify --bgp 59715.bgp --rir 59715.rir
```

Excellent work, what declared in the RIR DB matches what was found on the Internet (and vice versa)

Questo significa che la verifica ha dato esito positivo, cioè il candidato ha superato la prima fase del flusso di verifica, potendo così accedere alla verifica delle altre due azioni MANRS.

8.3 DETTAGLIO DELLA FASE 5: VERIFICA DELL'AZIONE 1 DI MANRS

Si inizia controllando se il candidato è conforme alle regole *anti-spoofing*. Per prima cosa si considera la configurazione del *router* del candidato e la si usa come *input* per un *parser* appositamente sviluppato che restituisce le relazioni con altri sistemi autonomi tramite BGP. A questo punto si crea uno scenario all'interno di *kathará* che emula una piccola Internet.

kathará può creare tanti *router* quanti sono i sistemi autonomi che hanno rapporti con il candidato, siano essi di transito, di *peering* o clienti. I *router* emulati da *kathará* sono basati sul *software FRRouting* e vengono creati automaticamente in base alla configurazione del candidato. Per testare le regole *anti-spoofing* viene messa a punto questa procedura: dalle reti appartenenti al candidato vengono estratti indirizzi IP casuali (IPv4 e IPv6) e assegnati a un *client* basato su kernel Linux emulato su *kathará*. Si attiva quindi un comando *ping* nel *client* per trasmettere pacchetti ICMP (*Internet Control Message Protocol*, *type 8 code 0*) che hanno come sorgente un IP *spoofed* e come destinazione un IP di un altro *client*. Su quest'ultimo *computer* è in esecuzione un *tcpdump* [112] che segnala l'arrivo dei pacchetti ICMP. Quindi, se il pacchetto sorgente *spoofed* arriva a destinazione, significa che o non ci sono regole *anti-spoofing* nella configurazione del candidato o, se ci sono, non funzionano. Per la riuscita del test, è sufficiente che un solo IP *spoofed* riesca a transitare verso il dispositivo della vittima dato che l'impostazione della regola sui *router*, quando c'è, esplicita soltanto i prefissi originanti localmente, lasciando così tutto il resto dello spazio di indirizzamento (IPv4 o IPv6) escluso.

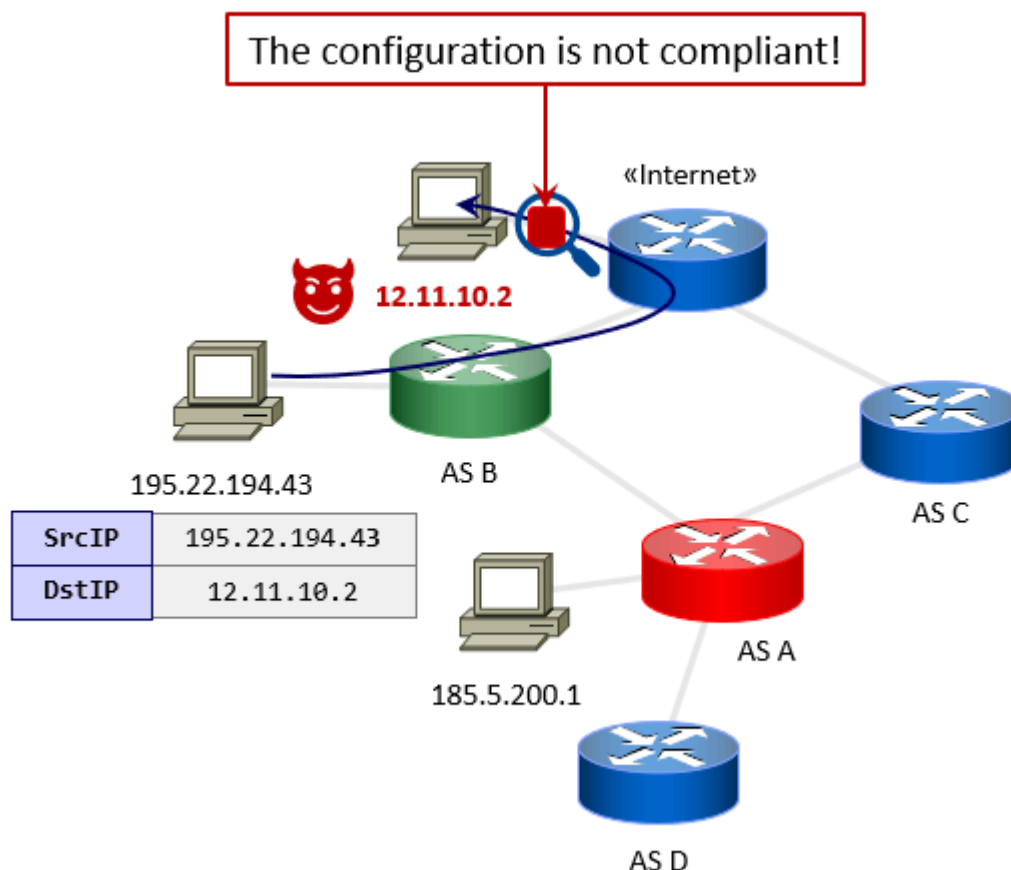


Fig. 4: Non conformità della configurazione all'azione 1 di MANRS

In caso contrario, il sistema non ha rilevato comportamenti anomali, suggerendo una possibile conformità all'azione 1 di MANRS, pur nei limiti del *test* effettuato.

8.3.1 Nota metodologica sulla distinzione tra *test* e verifica

È importante sottolineare che quanto descritto nella procedura di validazione dell'azione 1 costituisce un *test*, e non una verifica formale. Il *test* ha lo scopo di rilevare configurazioni inadeguate, attraverso l'invio controllato di pacchetti ICMP con indirizzo IP sorgente falsificato (*spoofed*).

Tuttavia, anche se nessun pacchetto *spoofed* riesce a superare il filtro, questo non consente di concludere con certezza che la configurazione sia completamente adeguata: è possibile che indirizzi IP non testati possano comunque transitare. In sintesi, il *test* può dimostrare la presenza di un problema, ma non può garantire l'assenza di ogni possibile falla. Una verifica formale, al contrario, richiederebbe un'analisi esaustiva della configurazione e dell'intero spazio di indirizzamento. RoSe-T quindi effettua un

test pratico ed efficace, utile per individuare gravi omissioni, ma non equivale a una dimostrazione formale di conformità completa.

8.4 DETTAGLIO DELLA FASE 5: VERIFICA DELL'AZIONE 2 DI MANRS

L'altro controllo da effettuare riguarda i filtri per le reti annunciate dai client del candidato. Cioè, il candidato deve assicurarsi di accettare solo le reti che i suoi clienti sono autorizzati ad annunciare. Per eseguire questo controllo, è necessario innanzitutto verificare nei *database* IRR le politiche di instradamento dichiarate dalle varie parti coinvolte nella configurazione appartenente al candidato. Poi su *kathará* si crea uno scenario in grado di emulare la relazione che esiste tra il sistema autonomo del candidato e i suoi clienti. A questo punto da uno dei *router* dei clienti (che ricordo sono emulati tramite *FRRouting*) si tenta di annunciare una rete estranea e si osserva se tale rete viene passata, ad esempio, al transito del candidato o ai suoi *peer* (anch'essi emulati su *kathará*). Se la rete estranea passa, significa che il candidato non ha stabilito regole efficaci; In caso contrario, l'assenza di prefissi non autorizzati nelle osservazioni disponibili suggerisce che il candidato possa essere conforme all'azione 2 di MANRS, pur tenendo conto dei limiti intrinseci del *test*.

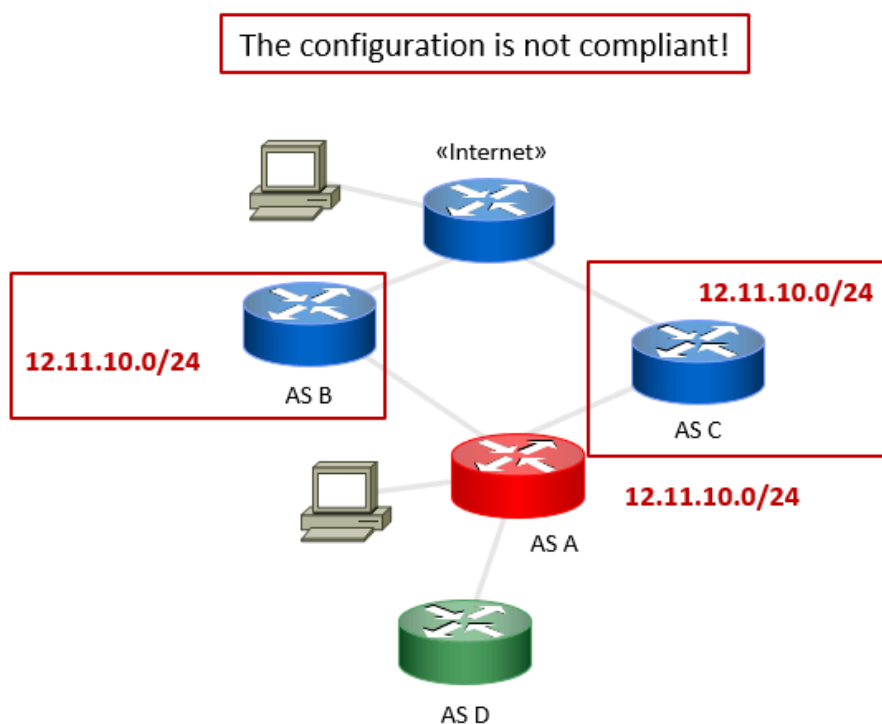


Fig. 5: Non conformità della configurazione all'azione 2 di MANRS

Anche in questo caso, per la riuscita del *test*, è sufficiente che una sola rete estranea venga annunciata verso Internet dato che l'impostazione della regola sui *router*, quando c'è, esplicita soltanto i prefissi originanti localmente (e quelli del *Customer-cone*), lasciando così tutto il resto dello spazio di indirizzamento (IPv4 o IPv6) escluso.

8.4.1 Nota metodologica sulla validazione dell'Azione 2

Anche in questo caso, il processo descritto costituisce un *test* osservazionale, che permette di individuare eventuali prefissi non autorizzati annunciati da un AS. Tuttavia, la sola assenza di annunci anomali in un certo intervallo temporale o su un certo punto di osservazione non garantisce la conformità completa. Infatti, un AS potrebbe annunciare prefissi non autorizzati in altri momenti, su altri *peer* o con meccanismi transitori non visibili al *test*. La conformità all'Azione 2 può quindi essere ragionevolmente inferita quando il *test* non rileva irregolarità, ma non può essere formalmente dimostrata senza una verifica globale e continua.

8.5 ROSE-T: SUPPORTO DEI VENDOR

RoSe-T supporta le configurazioni prodotte dai *router* dei seguenti fornitori:

- **Juniper VMX** (≥ 18.2), attualmente sono supportate solo le configurazioni *flat* cioè dove tutti i dispositivi si trovano all'interno di un'unica sottorete o VLAN, senza segregazione o suddivisione in più reti logiche separate, attraverso una VM `hellt/vrnetlab` [\[76\]](#) incorporata in un contenitore Docker. Si fa uso di una versione personalizzata della VM. [\[77\]](#)
- **Cisco IOS XR** ($\geq 7.9.2$) utilizzando l'immagine Docker ufficiale del piano di controllo XRd. È necessario configurare correttamente il calcolatore *host* prima di eseguire il contenitore XRd [\[78\]](#). In particolare, è necessario aumentare `fs.inotify.max_user_instances`

```
sysctl -w fs.inotify.max_user_instances=64000
```

e `fs.inotify.max_user_watches` ad almeno 64000

```
sysctl -w fs.inotify.max_user_watches=64000
```

- **MikroTik RouterOS** (≥ 7.16), al momento si accettano solo le configurazioni *non-terse* (cioè, non esportate con il parametro *terse*), attraverso una VM

hellt/vrnetlab incorporata in un contenitore Docker. Si fa uso di una versione personalizzata della VM [\[77\]](#).

9 CONCLUSIONI

Nonostante questo viaggio, lungo e impegnativo, sia giunto ormai al suo termine con la pubblicazione su GitHub [104] del metodo e del *software* elaborati durante i trascorsi tre anni, con la loro presentazione in congressi internazionali (Congresso RIPE87 a Roma nel 2023 [99]; Congresso ANIX 2024 [100], a Tirana) e nazionali (*Workshop* presso FT Winery Tour 2023 [101], a Calci PI; Terza Maratona Assoprovider [102] nel 2024) e con le esperienze degli operatori di rete che ne hanno fatto uso, ho potuto raccogliere alcuni interessanti stimoli per rendere ancor più funzionale il progetto.

Innanzitutto la necessità di supportare la sintassi di ulteriori *vendor* oltre quelli già contemplati (Cisco, Juniper, Mikrotik), poi l'opzione di elaborare le configurazioni di molteplici *router* contemporaneamente. Inoltre, d'accordo con *Global Cyber Alliance* che oggi gestisce l'iniziativa MANRS, l'evoluzione desiderata di RoSe-T è quella di saper elaborare, oltre il programma per gli operatori di rete già disponibile, anche le azioni per altri protagonisti di Internet come i punti di interscambio (IXP) e i distributori di contenuti (CDN); parallelamente la richiesta di GCA è di cominciare a lavorare per supportare la versione più evoluta dell'iniziativa MANRS che si chiama MANRS+ [116].

Questa forte spinta proviene innegabilmente anche dalle previsioni normative europee sulla sicurezza contenute dalla cosiddetta NIS2 [114] dove, a proposito del *routing*, si legge che è necessario “applicare le migliori pratiche per la sicurezza del DNS e per la sicurezza e l'igiene dell'instradamento in internet del traffico proveniente dalla rete e ad essa destinato” [115]. In tal senso la sicurezza del *routing* si pone come elemento cruciale per la stabilità dell'economia digitale di tutto il mondo. È stato dimostrato infatti anche in questa tesi che incidenti come i dirottamenti del BGP possono causare perdite economiche significative, interruzioni di servizi critici e, in genere, danni per le aziende. Attraverso strumenti come RoSe-T, è possibile non solo prevenire tali incidenti, ma anche, in modo indiretto, rafforzare la fiducia degli utenti finali nell'ecosistema di Internet.

Il contributo innovativo che questo lavoro intende apportare sia alla comunità accademica sia all'industria di Internet consiste nell'introduzione di un modello pratico per affrontare le vulnerabilità del BGP, cioè nella verifica automatizzata tramite l'emulazione allo scopo di certificare gli operatori di rete nel rispetto delle azioni proposte dal programma MANRS.

In futuro, tra i numerosi *fork* che tale lavoro consente, sarà necessario lavorare su una integrazione tra l'intelligenza artificiale e RoSe-T, così da spingere il paradigma dell'emulazione, in questa tesi dimostrato, sino a una completa automazione delle verifiche, da una parte, e a una piena comprensione di tutti i linguaggi implementati dai *vendor* nei dispositivi di *routing* dall'altra, persino uscendo dal perimetro del programma MANRS e ampliando la superficie delle configurazioni da verificare.

A tal fine, uno sviluppo strategico prevederà l'adozione di tecniche basate su modelli di apprendimento automatico supervisionati e non supervisionati, in grado di analizzare configurazioni esistenti, identificarne schemi ricorrenti e suggerire ottimizzazioni o anomalie. Un ulteriore filone di ricerca sarà orientato all'impiego di modelli linguistici (LLM) in grado di interpretare e tradurre configurazioni eterogenee dei diversi *vendor* verso una rappresentazione intermedia normalizzata, utile tanto per l'automazione quanto per la verifica semantica. Al momento in letteratura il più recente lavoro che si fonda sull'intelligenza artificiale per il miglioramento della comprensione delle configurazioni BGP è *DeepBGP* [103], un sistema basato su reti neurali e tecniche evolutive per generare automaticamente configurazioni BGP a partire da requisiti di alto livello definiti dall'operatore di rete. L'integrazione tra RoSe-T e *DeepBGP* consentirebbe di coprire l'intero ciclo di vita delle configurazioni BGP: dalla generazione, alla verifica, fino all'emulazione. La possibilità che viene data all'operatore sarebbe di definire un intento, cioè per esempio la conformità a MANRS, oppure le preferenze di uscita su Internet, la resilienza, ecc. Questo intento verrebbe quindi sottoposto a *DeepBGP* che sintetizza automaticamente le configurazioni per *router* Huawei/Cisco come attualmente supportato. Successivamente RoSe-T importerebbe queste configurazioni, le verificherebbe semanticamente e preparerebbe una emulazione da far girare su *katharà*. Se emergessero comportamenti inattesi o configurazioni non aderenti al comportamento voluto, RoSe-T dovrebbe essere in grado di generare *report* strutturati che possano essere utilizzati per aggiornare il *training set* di *DeepBGP*. Si creerebbe così un *loop* di miglioramento continuo, in cui *DeepBGP* genera e RoSe-T verifica e affina.

Un elemento fondamentale emerso da questo lavoro è la distinzione tra *test* e verifica: RoSe-T realizza controlli pratici e osservabili, capaci di rilevare anomalie manifeste, ma non può ancora fornire una garanzia formale di conformità. L'obiettivo a lungo termine è evolvere verso forme di verifica automatizzata e semantica, che integrino osservazione dinamica e comprensione profonda delle configurazioni.

In questa prospettiva, la consapevolezza dei limiti del *test* non rappresenta un ostacolo, bensì una base solida su cui costruire un metodo rigoroso. La sfida che resta aperta è trasformare questa consapevolezza in valore operativo, gettando le fondamenta per una nuova generazione di strumenti intelligenti, trasparenti e verificabili.

10 APPENDICE A

10.1 I COMANDI PER LA VERIFICA DELLE AZIONI MANRS 1

E 2

Prima di cominciare è necessario clonare il progetto RoSe-T da GitHub:

```
git clone https://github.com/OpenAccess-Italia/roset.git
```

Poi preparare l'ambiente virtuale per l'esecuzione di python e installare tutti i programmi che costituiscono i requisiti di RoSe-T:

```
cd roset
python3 -m venv ./venv
source ./venv/bin/activate
python -m pip install -r src/requirements.txt
cd resources
```

Ottenere una tabella di *routing* contenente la FIRT ad esempio dal RIR RIPE NCC:

```
wget
https://data.ris.ripe.net/rrc00/2024.12/updates.20241213.1305.
gz
```

A questo punto trasferiamo la FIRT in un nostro *database*:

```
python3 load_mrt.py updates.20241213.1305.gz rib_latest.db
cd ..
```

Arricchiamo l'ambiente con il *software* VR NET LAB che facilita la virtualizzazione dei *router*, poi applichiamo le nostre *patch* per farlo funzionare con *kathará*:

```
git clone https://github.com/hellt/vrnetlab.git
cd vrnetlab
git apply ../vrnet_patches/vrnet.patch
```

È il momento quindi di ottenere il *software* del *router* da virtualizzare, in questo caso il Routeros7 di Mikrotik:

```
cd routeros
wget
https://download.mikrotik.com/routeros/7.17rc3/chr-7.17rc3.vmd
k.zip
unzip chr-7.17rc3.vmdk.zip
```

Assicurarsi che *docker* sia attivo sulla macchina e lanciare la creazione del nostro ambiente di lavoro:

```
make
```

Infine è il momento di eseguire la verifica della configurazione, sia per l'azione 1 sia per l'azione 2 di MANRS.

```
sudo -E PATH=$PATH python test.py --config_path  
../candidato/routeros7.cfg --config_syntax Routeros
```

L'*output* di questo comando consentirà di conoscere il risultato della verifica, cioè se il sistema autonomo candidato riesca a filtrare gli IP *spoofed* e se riesca a filtrare correttamente le reti degli eventuali clienti.

10.2 I COMANDI PER LA VERIFICA DELL'AZIONE MANRS 4

Innanzitutto è necessario scaricare dagli archivi di *Route Views* l'ultima tabella disponibile:

```
wget
```

```
https://archive.routeviews.org/bgpdata/2024.12/RIBS/rib.202412  
13.1000.bz2
```

Estrarre dalla tabella le righe che contengono gli AS PATH che terminano con il sistema autonomo del candidato e popolare, con il risultato, il file nominato *\$CANDIDATO.rir*:

```
bgpgrep rib.20241213.1000.bz2 -aspath "59715$">59715.bgp
```

Leggere dall'IRR di riferimento l'oggetto *aut-num*: così da salvare nel file *\$CANDIDATO.rir* la politica di instradamento pubblicata:

```
whois -h whois.ripe.net AS59715|sed -n  
'/aut-num/,/source/p'>59715.rir
```

Eseguire il comando *Prolog* dando come input i due file precedentemente creati così da confrontarne il contenuto:

```
./verify --bgp 59715.bgp --rir 59715.rir
```

Excellent work, what declared in the RIR DB matches what was found on the Internet (and vice versa).

In questo caso la verifica è andata a buon fine, cioè quanto dichiarato negli IRR corrisponde a quanto si può osservare su Internet e viceversa. Si può tentare a questo punto una ulteriore verifica prendendo in considerazione un altro sistema autonomo.

```
bgpgrep rib.20241213.1000.bz2 -aspath "12874$">12874.bgp
```

```
whois -h whois.ripe.net AS12874|sed -n  
'/aut-num/,/source/p'>12874.rir
```

```
./verify --bgp 12874.bgp --rir 12874.rir  
Transit for AS12874 detected on the Internet but not declared  
in the RIR DB.
```

```
AS3741  
AS6461  
AS6774  
AS9002  
AS34224  
AS57463
```

```
Transit for AS12874 declared in the RIR DB but not detected in  
the Internet.
```

```
AS702  
AS1755  
AS3257  
AS3549  
AS3651  
AS60501  
AS202193  
AS202339
```

Come si può osservare, in questo caso la verifica non va a buon fine e dunque il flusso di lavoro di RoSe-T si interrompe fino a che questo passaggio non superi l'esame.

11 APPENDICE B

11.1 DOCUMENTAZIONE DEI PREDICATI USATI NEL CAPITOLO 6

Predicato	Arità	Descrizione
<code>provider(AS1, AS2)</code>	2	AS1 è il <i>provider</i> di AS2.
<code>neighbor(AS1, AS2)</code>	2	Sessione BGP attiva tra AS1 e AS2.
<code>route(AS, Prefix, Route)</code>	3	Percorso <i>Route</i> verso <i>Prefix</i> noto all'AS.
<code>route(Sender, Receiver, Prefix, Route)</code>	4	<i>Sender</i> annuncia a <i>Receiver</i> il percorso <i>Route</i> per <i>Prefix</i> .
<code>route_in_table(AS, Table, Prefix)</code>	3	La tabella <i>Table</i> dell'AS contiene instradamenti verso <i>Prefix</i> .
<code>route_in_table(AS, Table, Peer)</code>	3	La tabella <i>Table</i> dell'AS contiene instradamenti appresi da <i>Peer</i> .
<code>firt(Prefix)</code>	1	<i>Prefix</i> appartiene alla <i>Full Internet Routing Table</i> .
<code>as1_set(AS, List)</code>	2	Prefissi annunciati dal sistema autonomo AS (ad esempio, as1).
<code>as2_set(AS, List)</code>	2	Prefissi annunciati dal sistema autonomo AS (ad esempio, as2).
<code>not_provider(AS)</code>	1	L'AS non ha percorsi verso almeno un prefisso nella FIRT.

<code>compute_route(AS, Prefix, Route)</code>	3	Simulazione del calcolo di un percorso per un prefisso.
<code>member(X, List)</code>	2	Predicato <i>standard</i> : X è elemento della lista List.
<code>next_hop(Route, AS)</code>	2	AS è il <i>next-hop</i> del percorso <i>Route</i> .
<code>prefix(Route, Prefix)</code>	2	Il percorso <i>Route</i> termina con il prefisso <i>Prefix</i> .

12 BIBLIOGRAFIA

- [1] Y. Rekhter and T. Li, “A Border Gateway Protocol 4 (BGP-4),” RFC Editor, Mar. 1995. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc1771>
- [2] Y. Rekhter and T. Li, “A Border Gateway Protocol 4 (BGP-4),” RFC Editor, Jan. 2006. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc4271>
- [3] P. Traina, “BGP-4 Protocol Analysis,” RFC Editor, Mar. 1995. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc1774>
- [4] V. Fuller, T. Li, J. Yu, and K. Varadhan, “Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy,” RFC Editor, Sep. 1993. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc1519>
- [5] C. Timberg, “Quick fix for an early Internet problem lives on a quarter-century later,” Washington Post. Accessed: Nov. 15, 2024. [Online]. Available: <https://www.washingtonpost.com/sf/business/2015/05/31/net-of-insecurity-part-2/>
- [6] “Autonomous System (AS) Number Assignment Policies,” RIPE Network Coordination Center. Accessed: Nov. 15, 2024. [Online]. Available: <https://www.ripe.net/publications/docs/ripe-679>
- [7] J. Hawkinson and T. Bates, “Guidelines for creation, selection, and registration of an Autonomous System (AS),” RFC Editor, Mar. 1996. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc1930>
- [8] J. Haas and J. Mitchell, “Reservation of Last Autonomous System (AS) Numbers,” RFC Editor, Jul. 2014. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc7300>
- [9] W. Kumari, R. Bush, H. Schiller, and K. Patel, “Codification of AS 0 Processing,” RFC Editor, Aug. 2015. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc7607>
- [10] G. Huston, “Autonomous System (AS) Number Reservation for Documentation Use,” RFC Editor, Dec. 2008. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc5398>
- [11] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, and E. Lear, “Address Allocation for Private Internets,” RFC Editor, Feb. 1996. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc1918>
- [12] J. Abley and W. Sotomayor, “AS112 Nameserver Operations,” RFC Editor, May 2015. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc7534>

- [13] Q. Vohra and E. Chen, “BGP Support for Four-Octet Autonomous System (AS) Number Space,” RFC Editor, Dec. 2012. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc6793>
- [14] Q. Vohra and E. Chen, “BGP Support for Four-octet AS Number Space,” RFC Editor, May 2007. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc4893>
- [15] J. Mitchell, “Autonomous System (AS) Reservation for Private Use,” RFC Editor, Jul. 2013. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc6996>
- [16] G. Huston and G. Michaelson, “Textual Representation of Autonomous System (AS) Numbers,” RFC Editor, Dec. 2008. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc5396>
- [17] G. Malkin, “RIP Version 2,” RFC Editor, Nov. 1998. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc2453>
- [18] D. Savage, J. Ng, S. Moore, D. Slice, P. Paluch, and R. White, “Cisco’s Enhanced Interior Gateway Routing Protocol (EIGRP),” RFC Editor, May 2016. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc7868>
- [19] J. Moy, “OSPF Version 2,” RFC Editor, Apr. 1998. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc2328>
- [20] R. Coltun, D. Ferguson, J. Moy, and A. Lindem, “OSPF for IPv6,” RFC Editor, Jul. 2008. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc5340>
- [21] “Open Systems Interconnection: protocol for providing the connectionless-mode network service (ISO/DIS 8473 and ISO/DIS 8348 DAD 1),” doi: 10.3403/00010826u.
- [22] Lixin Gao, “On inferring autonomous system relationships in the Internet,” *IEEE/ACM Transactions on Networking*, vol. 9, no. 6, pp. 733–745, 2001, doi: 10.1109/90.974527.
- [23] R. Govindan and A. Reddy, “An analysis of Internet inter-domain topology and route stability,” in *Proceedings of INFOCOM '97*, IEEE Comput. Soc. Press, pp. 850–857. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.1109/infcom.1997.644557>
- [24] G. Huston, “Interconnection, Peering, and Settlements,” www.isoc.org. Accessed: Nov. 15, 2024. [Online]. Available: https://web.archive.org/web/20080503232459/http://www.isoc.org/isoc/conferences/inet/99/proceedings/1e/1e_1.htm
- [25] N. Chatzis, G. Smaragdakis, and A. Feldmann, “On the importance of Internet eXchange Points for today’s Internet ecosystem,” arXiv.org. [Online]. Available: <http://arxiv.org/abs/1307.5264>

- [26] E. Jasinska, N. Hilliard, R. Raszuk, and N. Bakker, “Internet Exchange BGP Route Server,” RFC Editor, Sep. 2016. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc7947>
- [27] N. Hilliard, E. Jasinska, R. Raszuk, and N. Bakker, “Internet Exchange BGP Route Server Operations,” RFC Editor, Sep. 2016. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc7948>
- [28] V. Fuller, T. Li, J. Yu, and K. Varadhan, “Supernetting: an Address Assignment and Aggregation Strategy,” RFC Editor, Jun. 1992. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc1338>
- [29] V. Fuller, T. Li, J. Yu, and K. Varadhan, “Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy,” RFC Editor, Sep. 1993. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc1519>
- [30] C. Partridge, T. Mendez, and W. Milliken, “Host Anycasting Service,” RFC Editor, Nov. 1993. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc1546>
- [31] H. Ballani and P. Francis, “Towards a global IP anycast service,” *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 4, pp. 301–312, Aug. 2005, doi: 10.1145/1090191.1080127.
- [32] “Describe Administrative Distance,” Cisco. Accessed: Nov. 15, 2024. [Online]. Available: <https://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/15986-admin-distance.html>
- [33] “Select BGP Best Path Algorithm,” Cisco. Accessed: Nov. 15, 2024. [Online]. Available: <https://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/13753-25.html>
- [34] “Understanding BGP Path Selection,” Juniper Networks. Accessed: Nov. 15, 2024. [Online]. Available: https://www.juniper.net/documentation/en_US/junos/topics/reference/general/routing-protocols-address-representation.html
- [35] “BGP Route Selection,” Huawei. Accessed: Nov. 15, 2024. [Online]. Available: <https://support.huawei.com/enterprise/en/doc/EDOC1100055099/70e83769/bgp-route-selection-rules>
- [36] S. Cheshire, B. Aboba, and E. Guttman, “Dynamic Configuration of IPv4 Link-Local Addresses,” RFC Editor, May 2005. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc3927>

- [37] R. Hinden and B. Haberman, "Unique Local IPv6 Unicast Addresses," RFC Editor, Oct. 2005. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc4193>
- [38] R. Hinden and S. Deering, "IP Version 6 Addressing Architecture," RFC Editor, Feb. 2006. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc4291>
- [39] G. Huston and N. Buraglio, "Expanding the IPv6 Documentation Space," RFC Editor, Aug. 2024. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc9637>
- [40] IANA, "Special-Use IPv4 Addresses," RFC Editor, Sep. 2002. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc3330>
- [41] G. Jones, "Operational Security Requirements for Large Internet Service Provider (ISP) IP Network Infrastructure," RFC Editor, Sep. 2004. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc3871>
- [42] A. Retana, R. White, V. Fuller, and D. McPherson, "Using 31-Bit Prefixes on IPv4 Point-to-Point Links," RFC Editor, Dec. 2000. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc3021>
- [43] "Local Preference for BGP Routes," Juniper Networks. Accessed: Nov. 15, 2024. [Online]. Available: https://www.juniper.net/documentation/en_US/junos/topics/topic-map/bgp-local-preference.html
- [44] "Connecting to a Service Provider Using External BGP," Cisco. Accessed: Nov. 15, 2024. [Online]. Available: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/iproute_bgp/configuration/15-mt/irg-15-mt-book/irg-external-sp.html
- [45] "Default Local Preference," Huawei. Accessed: Nov. 15, 2024. [Online]. Available: <https://support.huawei.com/enterprise/en/doc/EDOC1100127035/68702eed/default-local-preference>
- [46] K. Lougheed and Y. Rekhter, "Border Gateway Protocol (BGP)," RFC Editor, Jun. 1990. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc1163>
- [47] K. Lougheed and Y. Rekhter, "Border Gateway Protocol 3 (BGP-3)," RFC Editor, Oct. 1991. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc1267>
- [48] Y. Rekhter, "A Border Gateway Protocol 4 (BGP-4)," RFC Editor, Jul. 1994. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc1654>
- [49] J. Dong, M. Chen, and A. Suryanarayana, "Subcodes for BGP Finite State Machine Error," RFC Editor, May 2012. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc6608>

- [50] K. Lougheed and Y. Rekhter, “Border Gateway Protocol (BGP),” RFC Editor, Jun. 1989. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc1105>
- [51] D. Pei, “Finite State Machines for BGP,” *UCLA CSD Technical Report*, 2004.
- [52] S. Narain, “Reconfiguration Planning,” *Princeton University*, 2010.
- [53] S. Malik, “SAT Solvers: A Condensed History,” *Princeton University*, 2010.
- [54] E. Al-Shaer, “Formal Methods In Networking,” *Princeton University*, 2010.
- [55] P. Zave, “Formal Methods in Networking,” *Princeton University*, 2010.
- [56] A. R. Voellmy, “Proof of an interdomain policy,” in *Proceedings of the 2nd ACM workshop on Assurable and usable security configuration*, New York, NY, USA: ACM, Nov. 2009, pp. 37–44. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.1145/1655062.1655071>
- [57] “How Cloudflare auto-mitigated a world record 3.8 Tbps DDoS attack,” The Cloudflare Blog. Accessed: Nov. 15, 2024. [Online]. Available: <https://blog.cloudflare.com/how-cloudflare-auto-mitigated-world-record-3-8-tbps-ddos-attack>
- [58] S. Kent, C. Lynn, and K. Seo, “Secure Border Gateway Protocol (S-BGP),” *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 4, pp. 582–592, Apr. 2000, doi: 10.1109/49.839934.
- [59] Q. Li, Y.-C. Hu, and X. Zhang, “Even Rockets Cannot Make Pigs Fly Sustainably: Can BGP be Secured with BGPsec?,” in *Proceedings 2014 Workshop on Security of Emerging Networking Technologies*, Reston, VA: Internet Society, 2014. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.14722/sent.2014.23001>
- [60] M. Lepinski and K. Sriram, “BGPsec Protocol Specification,” RFC Editor, Sep. 2017. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc8205>
- [61] M. Wählisch, O. Maennel, and T. C. Schmidt, “Towards detecting BGP route hijacking using the RPKI,” *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 4, pp. 103–104, Aug. 2012, doi: 10.1145/2377677.2377702.
- [62] M. Lepinski and S. Kent, “An Infrastructure to Support Secure Internet Routing,” RFC Editor, Feb. 2012. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc6480>
- [63] P. Sermpezis *et al.*, “ARTEMIS: Neutralizing BGP Hijacking within a Minute,” arXiv.org. [Online]. Available: <https://arxiv.org/abs/1801.01085>
- [64] V. Kotronis, “ARTEMIS: Neutralising BGP Hijacking Within a Minute,” RIPE Labs. Accessed: Nov. 15, 2024. [Online]. Available: https://labs.ripe.net/author/vasileios_kotronis/artemis-neutralising-bgp-hijacking-within-a-mi

nute/

- [65] V. Kotronis, “ARTEMIS: an Open-source Tool for Detecting BGP Prefix Hijacking in Real Time,” RIPE Labs. Accessed: Nov. 15, 2024. [Online]. Available: https://labs.ripe.net/author/vasileios_kotronis/artemis-an-open-source-tool-for-detecting-bgp-prefix-hijacking-in-real-time/
- [66] E. Biersack *et al.*, “Visual analytics for BGP monitoring and prefix hijacking identification,” *IEEE Network*, vol. 26, no. 6, pp. 33–39, Nov. 2012, doi: 10.1109/mnet.2012.6375891.
- [67] “History,” MANRS. Accessed: Nov. 15, 2024. [Online]. Available: <https://manrs.org/about/history/>
- [68] “MANRS,” GCA | Global Cyber Alliance | Working to Eradicate Cyber Risk. Accessed: Nov. 15, 2024. [Online]. Available: <https://globalcyberalliance.org/manrs/>
- [69] A. Azimov, E. Bogomazov, R. Bush, K. Patel, and K. Sriram, “Route Leak Prevention and Detection Using Roles in UPDATE and OPEN Messages,” RFC Editor, May 2022. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc9234>
- [70] “Autonomous System (AS) Number Assignment Policies,” RIPE Network Coordination Center. Accessed: Nov. 15, 2024. [Online]. Available: <https://www.ripe.net/publications/docs/ripe-679/>
- [71] R. Bush, “Marking Announcements to BGP Collectors,” IETF Datatracker. Accessed: Nov. 15, 2024. [Online]. Available: <https://datatracker.ietf.org/doc/draft-ymbk-grow-bgp-collector-communities/02/>
- [72] E. Jasinska, N. Hilliard, R. Raszuk, and N. Bakker, “Internet Exchange BGP Route Server,” RFC Editor, Sep. 2016. Accessed: Nov. 15, 2024. [Online]. Available: <http://dx.doi.org/10.17487/rfc7947>
- [73] bgp, “ubgpsuite,” The DoubleFourteen Code Forge. Accessed: Nov. 15, 2024. [Online]. Available: <https://git.doublefourteen.io/bgp/ubgpsuite>
- [74] L. Cogotti, “Few performance facts about bgpgrep,” The DoubleFourteen Code Forge. Accessed: Nov. 15, 2024. [Online]. Available: <https://doublefourteen.io/blog/bgpgrep-performance-facts/>
- [75] Kathará, “Kathará,” Lightweight Container-based Network Emulation System. Accessed: Nov. 15, 2024. [Online]. Available: <https://www.kathara.org/>
- [76] hellt, “GitHub - hellt/vrnetlab: Make VM-based Network OSes run in Containerlab,” GitHub. Accessed: Nov. 15, 2024. [Online]. Available: <https://github.com/hellt/vrnetlab>
- [77] OpenAccess-Italia, “roset/vrnet_patches at main · OpenAccess-Italia/roset,” GitHub.

Accessed: Nov. 15, 2024. [Online]. Available:

https://github.com/rose-tool/roset/tree/main/vrnet_patches

[78] “Setting up the Host Environment to run XRd,” Virtual Routing IOS XR

Documentation @xrdocs. Accessed: Nov. 15, 2024. [Online]. Available:

<https://xrdocs.io/virtual-routing/tutorials/2022-08-22-setting-up-host-environment-to-run-xrd>

[79] S. Mattarella, “Gazzetta Ufficiale,” *Gazzetta Ufficiale*. Accessed: Dec. 05, 2024.

[Online]. Available: <https://www.gazzettaufficiale.it/eli/id/2024/10/01/24G00155/SG>

[80] P. Paganini, “Russian telco Rostelecom hijacks traffic for IT giants, including

Google,” *Security Affairs*. Accessed: Dec. 12, 2024. [Online]. Available:

<https://securityaffairs.com/101134/security/rostelecom-telco-hijacks-internet-traffic.html>

[81] A. Mcconachie, “BGP Hijacker Steals Bitcoins,” *Internet Society*. Accessed: Dec. 12, 2024. [Online]. Available:

<https://www.internetsociety.org/blog/2014/08/bgp-hijacker-steals-bitcoins/>

[82] C. Huadmai, “Verification of routing policies by using model checking technique,” in

Proceedings of the 6th IEEE International Conference on Intelligent Data Acquisition and

Advanced Computing Systems, IEEE, Sep. 2011, pp. 711–716. Accessed: Dec. 12, 2024.

[Online]. Available: <https://doi.org/10.1109/idaacs.2011.6072863>

[83] S. Haeri, D. Kresic, and L. Trajkovic, “Probabilistic verification of BGP

convergence,” in *2011 19th IEEE International Conference on Network Protocols*, IEEE, Oct.

2011, pp. 127–128. Accessed: Dec. 12, 2024. [Online]. Available:

<https://doi.org/10.1109/icnp.2011.6089037>

[84] K. Weitz, D. Woos, E. Torlak, M. D. Ernst, A. Krishnamurthy, and Z. Tatlock,

“Scalable verification of border gateway protocol configurations with an SMT solver,” *ACM*

SIGPLAN Notices, vol. 51, no. 10, pp. 765–780, Oct. 2016, doi: 10.1145/3022671.2984012.

[85] R. Beckett, A. Gupta, R. Mahajan, and D. Walker, “A General Approach to Network

Configuration Verification,” in *Proceedings of the Conference of the ACM Special Interest*

Group on Data Communication, New York, NY, USA: ACM, Aug. 2017. Accessed: Dec. 12,

2024. [Online]. Available: <https://doi.org/10.1145/3098822.3098834>

[86] H. Mai, A. Khurshid, R. Agarwal, M. Caesar, P. B. Godfrey, and S. T. King,

“Debugging the data plane with anteater,” *ACM SIGCOMM Computer Communication*

Review, vol. 41, no. 4, pp. 290–301, Aug. 2011, doi: 10.1145/2043164.2018470.

[87] X. Shao, Z. Chen, D. Holcomb, and L. Gao, “Accelerating BGP Configuration

Verification Through Reducing Cycles in SMT Constraints,” *IEEE/ACM Transactions on*

Networking, vol. 30, no. 6, pp. 2493–2504, Dec. 2022, doi: 10.1109/tnet.2022.3176267.

- [88] OpenAccess-Italia, “GitHub - OpenAccess-Italia/roset: ROSE-T: ROuting Security Tool,” GitHub. Accessed: Dec. 12, 2024. [Online]. Available: <https://github.com/OpenAccess-Italia/roset>
- [89] J. F. Nash, “EQUILIBRIUM POINTS IN n -PERSON GAMES,” in *Classics in Game Theory*, Princeton University Press, 2020, pp. 3–4. Accessed: Jan. 16, 2025. [Online]. Available: <https://doi.org/10.2307/j.ctv173f1fh.6>
- [90] D. Fudenberg and J. Tirole, *Game Theory*. MIT Press, 1991.
- [91] E. M. Clarke, O. Grumberg, D. Peled, and D. A. Peled, *Model Checking*. MIT Press, 1999.
- [92] R. Beckett, A. Gupta, R. Mahajan, and D. Walker, “Abstract interpretation of distributed network control planes,” *Proceedings of the ACM on Programming Languages*, vol. 4, no. POPL, pp. 1–27, Dec. 2019, doi: 10.1145/3371110.
- [93] A. Fogel, “A General Approach to Network Configuration Analysis,” *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*, 2015.
- [94] N. P. Lopes and A. Rybalchenko, “Fast BGP Simulation of Large Datacenters,” in *Lecture Notes in Computer Science*, Cham: Springer International Publishing, 2019, pp. 386–408. Accessed: Jan. 16, 2025. [Online]. Available: https://doi.org/10.1007/978-3-030-11245-5_18
- [95] E. W. Dijkstra, “A note on two problems in connexion with graphs,” *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, Dec. 1959, doi: 10.1007/bf01386390.
- [96] S. Prabhu, “Plankton: Scalable network configuration verification through model checking,” *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20)*, 2020.
- [97] G. J. Holzmann, “The model checker SPIN,” *IEEE Transactions on Software Engineering*, vol. 23, no. 5, pp. 279–295, May 1997, doi: 10.1109/32.588521.
- [98] X. Qie and S. Narain, “Using service grammar to diagnose BGP configuration errors,” *Science of Computer Programming*, vol. 53, no. 2, pp. 125–141, Nov. 2004, doi: 10.1016/j.scico.2003.12.007.
- [99] K. Weitz, “Bagpipe: Verified BGP configuration checking,” *University of Washington Department of Computer Science and Engineering*, 2016.
- [100] M. Brown, A. Fogel, D. Halperin, V. Heorhiadi, R. Mahajan, and T. Millstein, “Lessons from the evolution of the Batfish configuration analysis tool,” in *Proceedings of the ACM SIGCOMM 2023 Conference*, New York, NY, USA: ACM, Sep. 2023, pp. 122–135. Accessed: Jan. 16, 2025. [Online]. Available: <https://doi.org/10.1145/3603269.3604866>

- [101] S. Ceri, G. Gottlob, and L. Tanca, *Logic Programming and Databases*. Springer Science & Business Media, 2012.
- [102] P. Zhang *et al.*, “Incremental Network Configuration Verification,” in *Proceedings of the 19th ACM Workshop on Hot Topics in Networks*, New York, NY, USA: ACM, Nov. 2020. Accessed: Jan. 16, 2025. [Online]. Available: <https://doi.org/10.1145/3422604.3425936>
- [103] A. Wang, P. Basu, B. T. Loo, and O. Sokolsky, “Declarative Network Verification,” in *Lecture Notes in Computer Science*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 61–75. Accessed: Jan. 16, 2025. [Online]. Available: https://doi.org/10.1007/978-3-540-92995-6_5
- [104] N. Lopes, “Checking beliefs in dynamic networks,” *USENIX Association*, 2015.
- [105] N. McKeown *et al.*, “OpenFlow,” *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, Mar. 2008, doi: 10.1145/1355734.1355746.
- [106] P. Bosshart *et al.*, “P4,” *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 87–95, Jul. 2014, doi: 10.1145/2656877.2656890.
- [107] N. McKeown, “Automatically verifying reachability and well-formedness in P4 Networks,” Sep. 2016.
- [108] “FRRouting.” Accessed: Jan. 20, 2025. [Online]. Available: <https://frrouting.org/>
- [109] “Webupdates,” RIPE Network Coordination Centre. Accessed: Jan. 20, 2025. [Online]. Available: <https://apps.db.ripe.net/db-web-ui/query?from=www&searchtext=AS-SBTAP>
- [110] L. Blunk, J. Damas, F. Parent, and A. Robachevsky, “Routing Policy Specification Language next generation (RPSLNg),” RFC Editor, Mar. 2005. Accessed: Jan. 20, 2025. [Online]. Available: <https://doi.org/10.17487/rfc4012>
- [111] R. Bush, “Marking Announcements to BGP Collectors,” IETF Datatracker. [Online]. Available: <https://datatracker.ietf.org/doc/html/draft-ymbk-grow-bgp-collector-communities-02>
- [112] TCPDUMP & LIBPCAP. Accessed: Jan. 22, 2025. [Online]. Available: <https://www.tcpdump.org/>
- [113] N. Giannarakis, D. Loehr, R. Beckett, and D. Walker, “NV: an intermediate language for verification of network control planes,” in *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation*, New York, NY, USA: ACM, Jun. 2020, pp. 958–973. Accessed: Apr. 02, 2025. [Online]. Available: <https://doi.org/10.1145/3385412.3386019>
- [114] “Consolidated text: Directive (EU) 2022/2555 of the European Parliament and of the

Council of 14 December 2022 on measures for a high common level of cybersecurity across the Union, amending Regulation (EU) No 910/2014 and Directive (EU) 2018/1972, and repealing Directive (EU) 2016/1148 (NIS 2 Directive) (Text with EEA relevance)Text with EEA relevance.” Accessed: Apr. 02, 2025. [Online]. Available:

<http://data.europa.eu/eli/dir/2022/2555/2022-12-27>

[115] “Commission Implementing Regulation (EU) 2024/2690 of 17 October 2024 laying down rules for the application of Directive (EU) 2022/2555 as regards technical and methodological requirements of cybersecurity risk-management measures and further specification of the cases in which an incident is considered to be significant with regard to DNS service providers, TLD name registries, cloud computing service providers, data centre service providers, content delivery network providers, managed service providers, managed security service providers, providers of online market places, of online search engines and of social networking services platforms, and trust service providers.” Accessed: Apr. 02, 2025. [Online]. Available: http://data.europa.eu/eli/reg_impl/2024/2690/oj

[116] “MANRS+ Concept,” MANRS. Accessed: Apr. 03, 2025. [Online]. Available: <https://manrs.org/about/manrs-working-group/manrs-concept/>