



Carnegie Mellon
Software Engineering Institute

CERT
Situational
Awareness

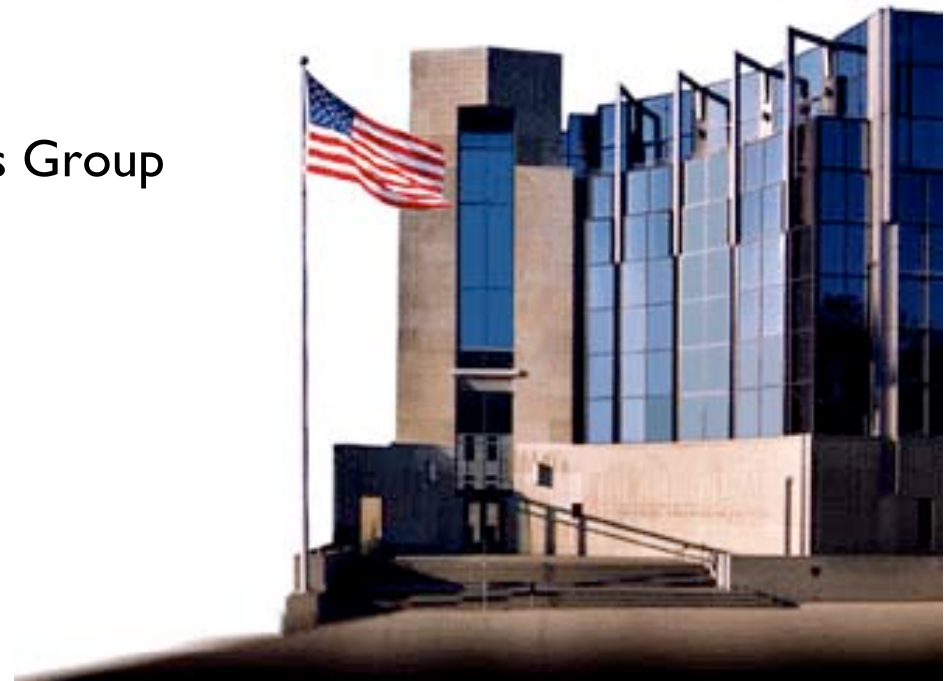
Preparing RIR Allocation Data for Network Security Analysis Tasks

Brian Trammell <bht@cert.org>

for NANOG 31, San Francisco, May 23-25, 2004

CERT® Network Situational Awareness Group
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213-3890

*The CERT Network Situational Awareness Group
is part of the Software Engineering Institute.
The Software Engineering Institute is sponsored by
the U.S. Department of Defense.*



Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE MAY 2004		2. REPORT TYPE		3. DATES COVERED 00-00-2004 to 00-00-2004	
4. TITLE AND SUBTITLE Preparing RIR Allocation Data for Network Security Analysis Tasks				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Carnegie Mellon University,Software Engineering Institute,Pittsburgh,PA,15213				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT Same as Report (SAR)	18. NUMBER OF PAGES 21	19a. NAME OF RESPONSIBLE PERSON
a. REPORT unclassified	b. ABSTRACT unclassified	c. THIS PAGE unclassified			

Background

- Actors in incident and traffic analysis data are expressed by IP address.
- Analysis operations categorize events at higher levels of abstraction:
 - by location
 - by organization
 - by network
- Correlation of incident and traffic analysis data with regional registry allocation data bridges the gap.

Requirements

- Group addresses into CIDR blocks.
- Provide network names, country codes, and POC handles for each block.
- Provide a tree view of IPv4 address space.
 - allows association of an address with a network at any level of allocation or assignment
 - we are not yet interested in IPv6 because we receive no incident or traffic data from IPv6 networks.
- Run periodically, require little operational support.
 - used for batch analysis
 - e.g., RIPE database mirror would be overkill

Data Sources

- Data from three RIRs:
 - ARIN (bulk data by request)
 - RIPE (publicly-available ripe.db.inetnum)
 - APNIC (publicly-available apnic.db.inetnum)
- 2,061,995 ranges (2,123,270 CIDR blocks) from these three sources (as of May 5, 2004)
- What about LACNIC?
 - LACNIC data omitted from counts pending bulk data request.

Assumptions and Anomalies

- Early in the effort, we assumed that...
 - the allocation tree is strictly a tree.
 - the registries agree on all allocations.
 - allocations and assignments are universally done in terms of CIDR blocks.
 - supplementary information (e.g., modification dates) are stored in a universally uniform format.
- >99.3% of ranges conform to these assumptions

Transformations

- Our tool chain (“AddrTree”) was designed to process allocation data from the RIRs for use in categorizing actors in incident and traffic data.
- AddrTree performs the following transformations:
 - Normalization of modification dates.
 - Elimination of redirect records.
 - Resolution of conflicts between regional registries.
 - Arrangement of allocations and assignments into a single tree structure.
 - Detection of anomalies in address ranges:
 - “Erosions” – off-by-one errors in allocation range ends.
 - “Inversions” – violations of tree hierarchy.
 - Splitting of allocation ranges into CIDR blocks.

Step 1: Parsing and Stripping

- Extracts essential information from registry text databases and transforms it into a compact, line-oriented text format
- Normalizes modification dates to ISO8601
 - Two-digit years (2,566)
 - YYYYDDMM date format (7)
 - only unambiguous instances of this anomaly can be detected.
 - Dates beyond end of month (4)
- Eliminates non-network (redirect) records (1,692)

Step 2: Merging

- Merges allocations between two registries into a single tree
 - Applied in stages to build a “world” allocation tree
- Detects and resolves conflicts between registries
 - most of these are early registrations
 - first by national affiliation (2,580)
 - the RIR with responsibility for the network’s country is probably more correct
 - then by registry seniority (3)
 - arbitrary, but avoids human intervention

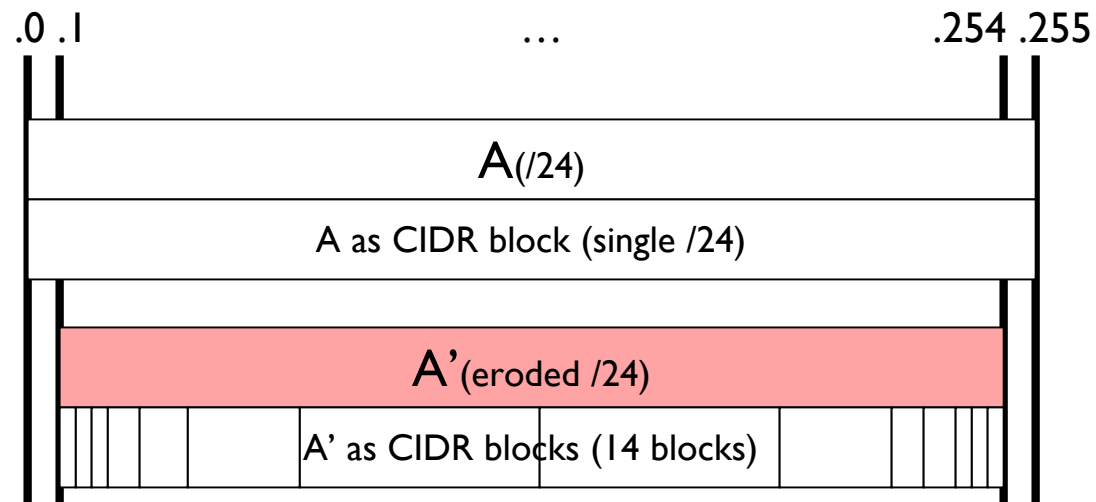
Step 3: Stacking and Splitting

- Range “erosions” are corrected before stacking.
- Stacking notes each record’s depth in the tree – necessary to maintain hierarchy of ranges after each range is split into CIDR blocks.
- Range “inversions” are detected during stacking.
- Each range record is replaced with one range per CIDR block covered by the record.
 - Growth in number of records is small.
 - > 99.98% of allocation ranges are a single block wide.

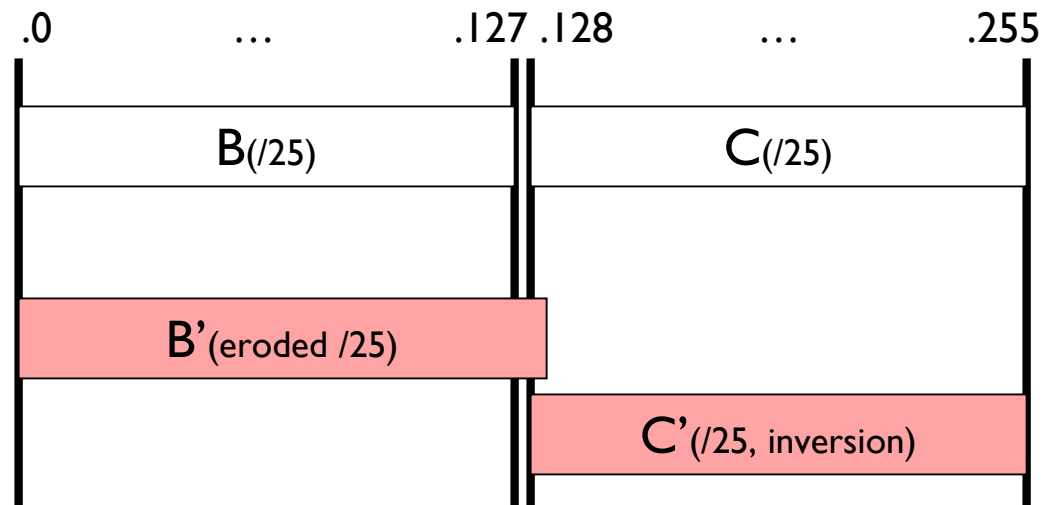
Range “Erosion”

- Off-by-one errors at start or end of a probable single CIDR block range
 - Missing network/broadcast address (“inner erosion”)
 - e.g., 10.2.3.0/24 as 10.2.3.1 - 10.2.3.254
 - End of range non-inclusive (“outer erosion”)
 - e.g., 10.2.3.0/25 as 10.2.3.0 - 10.2.3.128
- Inner erosions break CIDR block splitting.
 - 10.2.3.0/24 vs. 10.2.3.1/32, 10.2.3.2/31, 10.2.3.4/30, 10.2.3.8/29, 10.2.3.16/28, 10.2.3.32/27, 10.2.3.64/26, 10.2.3.128/26, 10.2.3.192/27, 10.2.3.224/28, 10.2.3.240/29, 10.2.3.248/30, 10.2.3.252/31, 10.2.3.254/32
- Outer erosions may cause inversions.

Inner Erosion Example



Outer Erosion Example



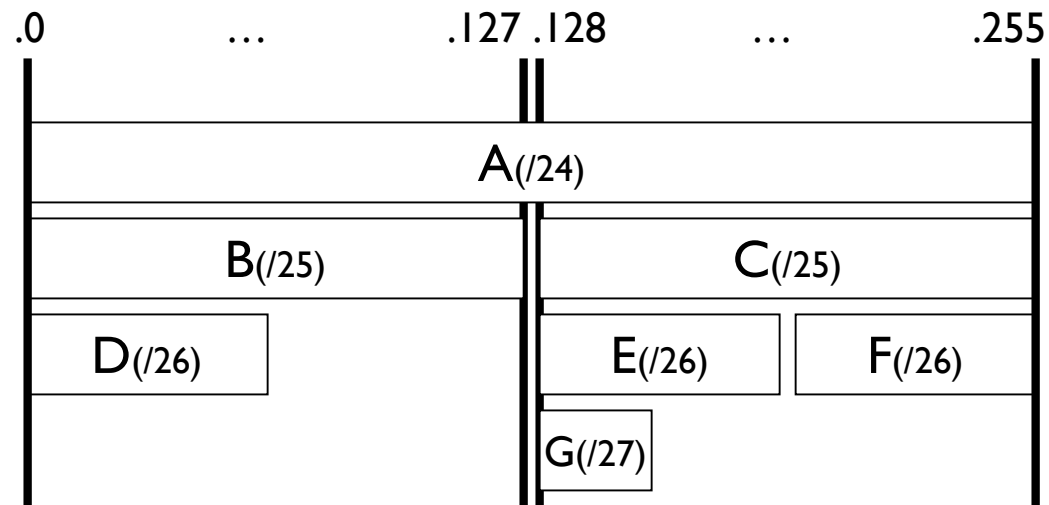
Erosion Statistics

- 6,404 range erosions detected:
 - 71.0% (4,547) are inner erosions.
 - 21.9% (1,400) are outer erosions.
 - 93% (1,302) of these are at end of range.
 - 7.1% (457) are “shifted” records (both outer and inner erosions).
- 98.9% on /24 blocks and smaller
- Erosions not detected on blocks smaller than /28.
 - Erosions on tiny blocks more likely to be false positives.

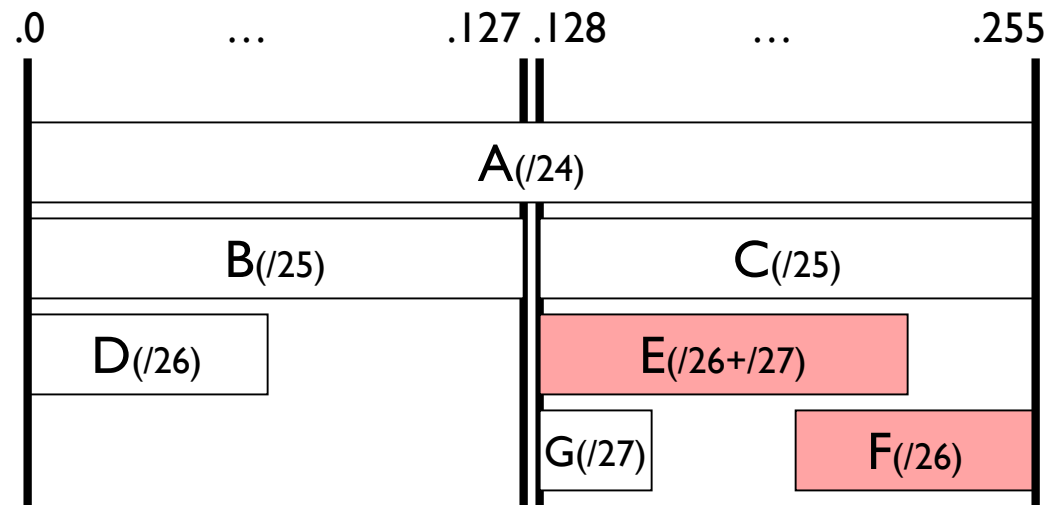
Range “Inversion”

- An inverted range is one which does not fit cleanly into the tree.
- May indicate errors in allocation ranges, or stale allocation records
- Difficult (if possible) to know which range in an inversion is correct
- Inversion correction is an area for future work.
 - Current procedure arbitrarily drops the second range.

Normal Allocation Tree



Inverted Allocation Tree



Inversion Statistics

- 745 range inversions detected:
 - typographical errors
 - e.g., 10.2.3.120 - 10.2.4.127 collides with 10.2.3.128-10.2.4.135; both are probably /29s, though it's not clear which /29s they should be.
 - simple overlap (possible stale records?)
 - outer erosions on tiny blocks
 - these are not fixed during erosion correction because of the higher risk of false positives on smaller blocks.
- Counts by type not available because inversion categorization is not yet automated.

Anomaly Logging

- Anomalies are logged during detection
 - for tuning of anomaly detection and correction techniques.
 - for transformation into a useful format for automated submission to the registries.

Future Work

- Inversion categorization
- Inversion correction
- Use of context to minimize false erosion detection
 - Allows erosion correction on tiny blocks
- Erosion detection on multiple-block ranges
- Automated anomaly submission to RIRs

For More Information

- <http://aircert.sourceforge.net/addrtree>

Appendix: AddrTree workflow

