

Issues and Techniques for Anonymizing Network Traces

Vern Paxson Ruoming Pang
ICSI/LBNL Princeton University

vern@icir.org/rpang@cs.princeton.edu

September 27, 2005

Impact of Anonymization on Research

- Major issue for soundly evaluating defense mechanisms: what sort of collateral damage do they incur?
 - ⇒ Requires “realistic” background traffic
- In addition, the problem of “Crud”
 - Real traffic is riddled with idiosyncratic/broken activity
- Some of this still doable w/ anon. data
 - ▶▶ Invaluable to have a *ground truth oracle*
 - ▶▶ And/or: *meta-data*

Impact of Lack of Traffic Contents on Research

- Much intrusion detection research requires packet contents
- Major intrusion detection research pitfall: failure to realistically assess false positives, particularly in how they scale to large networks
 - Especially for anomaly detection
- Lack of traffic contents increasingly worrisome as semantic level of attacks rises
- How can we leverage the need to know “what’s being said” but not “who is saying it”?

Fundamental Tension

- Research utility of traces diminishes as information is removed
- Especially matters for traces used as background traffic
- Question becomes: what are the *threats*, the data contributor's *threat model*, and the technology available for achieving the *policy* that incorporates these

Sensitive Information to Protect

- What does the infrastructure look like?
- How much talking is going on?
- Who is talking to whom?
- What are they saying?

- (The medical information disclosure viewpoint):
 - Identities
 - Confidential attributes

What Does the Infrastructure Look Like?

- Topology? Capacities? Hardware specifics? Future plans?
- Topology, capacities somewhat externally measurable. Others, only under NDA.
- Anonymize by constructing abstractions that the provider signs off on.
- If these abstractions are “good”, that works

How Much Talking is Going On?

- E.g., # customers, volume of traffic, how many web server hits
- Commercially sensitive
- Anonymized by expressing in purely relative terms
 - E.g., hourly fluctuation
- This is not broadly useful

Who is Talking to Whom?

- Usual approach: via 1-to-1 mapping of actual IDs to synthetic IDs
- Can be fully opaque or can partially preserve relationships, e.g.,
inside/outside, CIDR classes, `tcpdpriv`
-A50 (et al)
- Very common technique
- \exists inference attacks that leverage *structure*

What Are They Saying?

- Far and away most common answer: *You Don't Get To Know*
 - i.e., communication semantics completely stripped
- Or, if not, *syntactic* garbling: e.g., s/PASS
.*/PASS XXX/
 - ⇒ Can make inappropriate transformations
 - ⇒ Unsound if done on a per-packet basis

Attacks on Anonymization

- Inference attacks:
 - Fingerprinting via public/guessed info
 - E.g., file size/date \Rightarrow identity of file
 - E.g., software version, config \Rightarrow which server
 - E.g., HTTP item size \Rightarrow which item [*]

Attacks on Anonymization

- Dictionary attacks:
 - If known hash/scrambling function, cram zillions of candidates through it (or guess)
- Or: seed trace with known text
 - Look for its mapping, search for collisions
- E.g., insert “**RETR alice**” to find “**USER alice**”
- Counter-tactic: separate namespaces
 - E.g. Hash(file, server-ip, complete-path)

Attacks on Anonymization

- Structural attacks:
 - Consider a site trace w/ complete (but 1-to-1) rewriting of IP addresses
 - If from a typical site, then riddled with sequential scans
 - E.g., for an arbitrary day at LBNL:
 - Look for sequential scans of $\geq 10K$ addrs.
 - Find them from 59 different remote hosts
⇒ Can completely undo *any* 1→1 scrambling
 - Combat via $N \rightarrow 1$ scrambling
 - Or: identify scanners and remove them

Anonymization Tools

- Tcpsdpriv, Tcpspurify: strips transport payloads, rewrites addresses/ports to specified degree; noteworthy for “prefix-preserving” address rewrite mode
- Ipsumdump: extracts given fields from tcpdump trace, prints as ASCII
- Bro: *trace transformation* mode allows semantic rewriting of payloads

tcpmkpub

- Programmable/customizable trace anonymization
- Forces header field-by-field examination, *opt-in*
- Supports complex address transformation, crud retention, ***meta-data***
- Template driven, e.g:

```
FIELD          (TCP_SRCPORT,  2,      KEEP)
FIELD          (TCP_DSTPORT,  2,      KEEP)
FIELD          (TCP_SEQ,      4,      KEEP)
FIELD          (TCP_ACK,      4,      KEEP)
FIELD          (TCP_OFF,      1,      KEEP)
FIELD          (TCP_FLAGS,    1,      KEEP)
FIELD          (TCP_WINDOW,   2,      KEEP)
PUTOFF_FIELD   (TCP_CHKSUM,    2,      ZERO)
FIELD          (TCP_URGPTR,   2,      KEEP)
FIELD          (TCP_OPTIONS,  VARLEN, anonymize_tcp_options)
PICKUP_FIELD   (TCP_CHKSUM,    0,      recompute_tcp_checksum)
FIELD          (TCP_DATA,     RESTLEN, SKIP)
```

The Verification Problem

- How do we know that an anonymized trace is “safe”?
- Per [Pang/Paxson 2003]:
 - Use *filter-in* rather than *filter-out*
 - Anonymized trace only includes elements explicitly allowed
 - Fail-safe/conservative (white-lists, not black-lists)
 - But: find that *manual inspection* needed in order for trace to include “*crud*”
- Research on tools & principles needed

PREDICT-Specific Issues

- Researcher needs
 - Retain as much trace richness as possible
- Legal issues
 - Liability for what's exposed in datasets
- Perception issues
 - Danger for public misinterpretation of “DHS is gathering personal information”
- Threat model
 - PREDICT's vetting of repository users should help ease *some* data provider concerns