The SCRUB* Security Data Sharing Infrastructure

William Yurcik*
<byurcik@gmail.com>
Clay Woolam, Greg Hellings, Latifur Khan, Bhavani Thuraisingham
University of Texas at Dallas

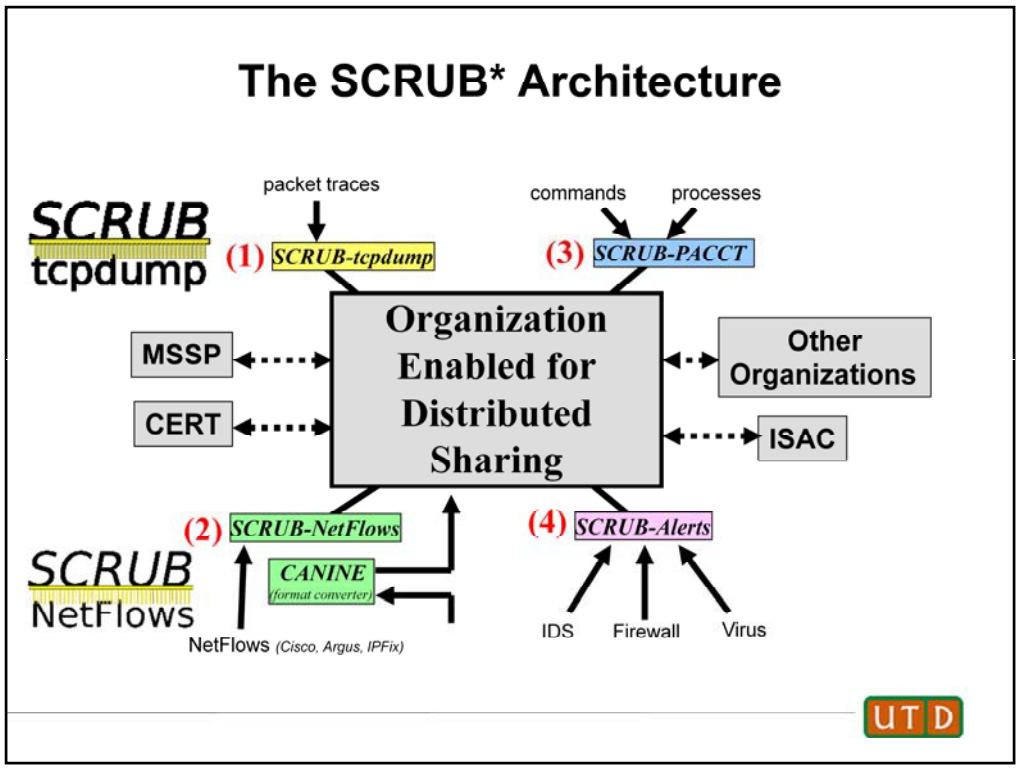
IEEE/IFIP Network Operations and Management Symposium (NOMS)
7-11 April 2008

Sharing data between organization is important aspect of network protection that is not currently occurring since it is unsafe.

This talk is about a suite of tools that can be used to "scrub" data (using anonymization) so it can be safely shared.

SCRUB* is an infrastructure because all the tools use the same anonymization algorithms for seamless sharing independent of the data source.

All authors are all affiliated with the University of Texas at Dallas.



There are four anonymization tools in the SCRUB tool suite.

(1) *SCRUB-tcpdump* for anonymizing packet traces

(2) *SCRUB-NetFlows* for anonymizating NetFlows network traffic logs

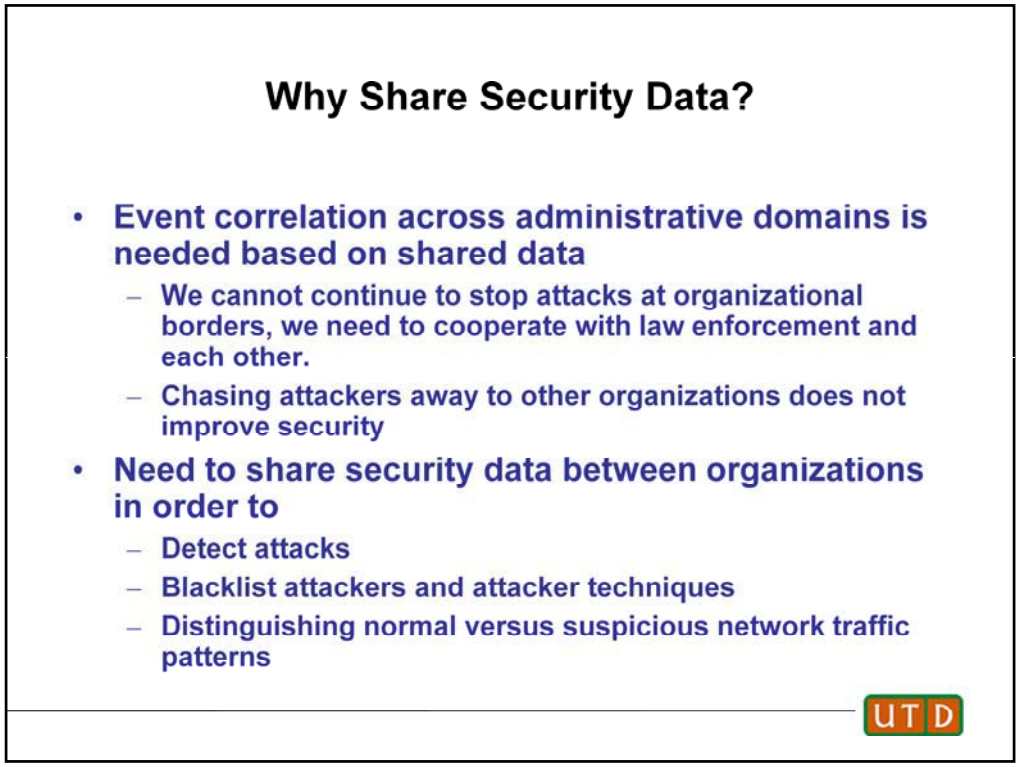
(3) *SCRUB-PACCT* for anonymizing process accounting

(4) *SCRUB-alerts* for anonymizing intrusion detection system alerts (or firewall or virus alerts)

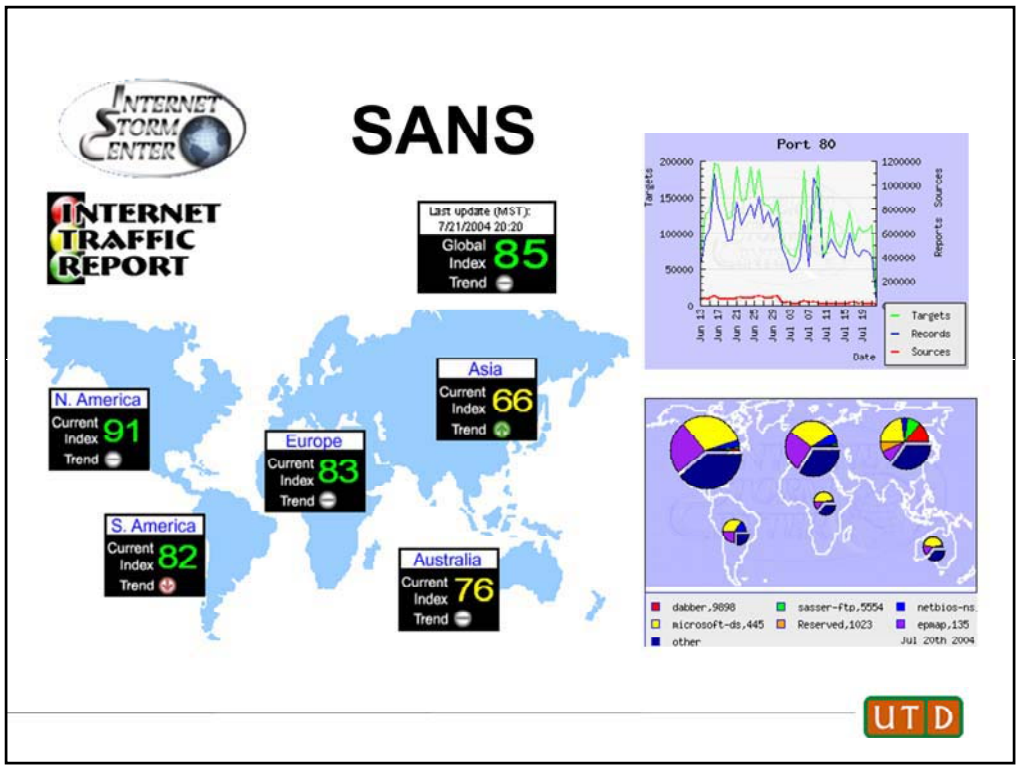
(5) Each of these tools will be described in more detail later in this presentation.

Packet traces, NetFlows, process accounting, and alerts make up the bulk of source data that is currently shared between organizations for security purposes, however, the SCRUB* architecture is extensible to include other anonymizers as needed.

An organization may share anonymized data with Other Organizations, government ISACs (Information Sharing and Analysis Centers), CERTs (Computer Emergency Response Teams), and/or MSSPs (Managed Security Service Providers).

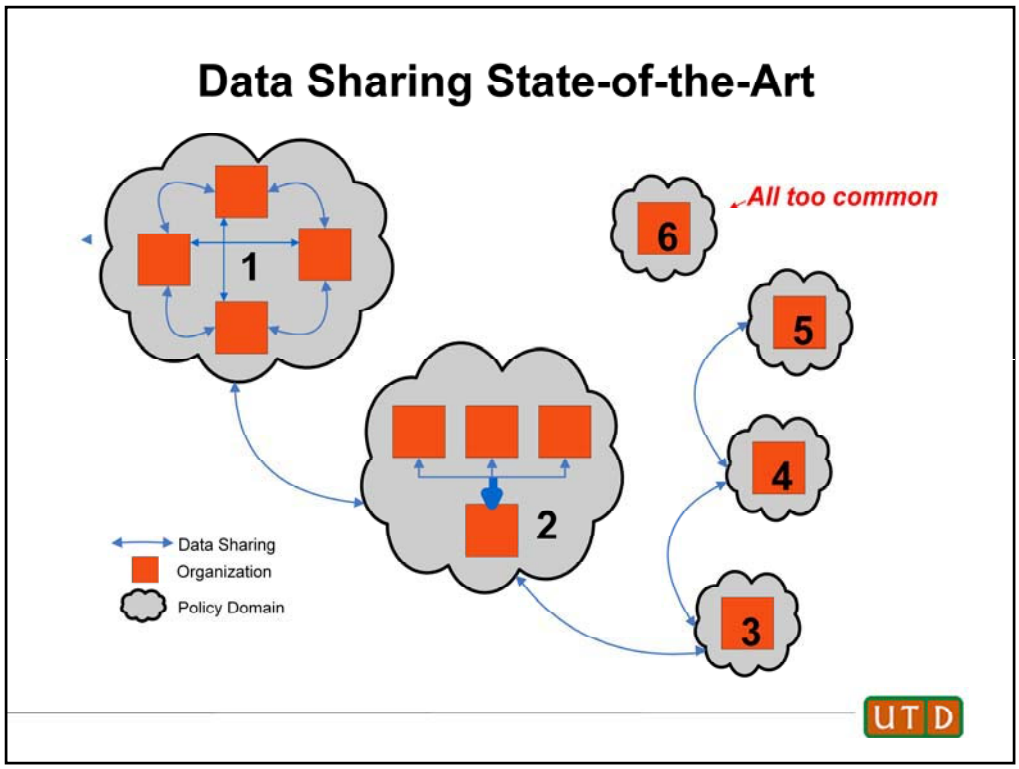


Security operations staff use data to help defend their own organizational networks. Since attackers typically attack across network boundaries and frequently change targets to attack within different security domains, effective protection requires defenders to look beyond their own organizational perimeter toward cooperation and data sharing with other organizations in order to defeat attackers. However, to date little or no data sharing has occurred between organizations due to practical concerns. Unfortunately, this is not also true for attackers who are quite efficient at sharing vulnerability and exploit information amongst themselves. Collaborative security analysis between organizations may include incident detection, trend analysis, attack detection, black listing specific attackers and attacker techniques, and distinguishing/modeling normal versus suspicious network traffic patterns.



The Internet Storm Center (ISC) <http://isc.sans.org/>

ISC is a grass-roots effort run by SANS. They collect IDS logs from volunteering organizations and analyze them to detect trends. Their purpose is to provide an early warning system of new worm activity on the Internet. They provide reports on the top ports being scanned with respect to time, and they use the trend information they find to determine the INFOCon threat level, much like Symantec defines the ThreatCon level with DeepSight data. ISC does not share actual logs, but they produce high level statistics from receiving shared logs. For this reason, their port activity and trends data do not need to be sanitized. They sanitize information about scanner source IPs by looking very broadly at the number of scans per class C network. This kind of anonymization is also used in other logs where they simply truncate IP addresses. The danger is pretty low in sharing this kind of information, but its usefulness is also minimal. It does nothing more than allow inferences such as “The US does the most scanning” or “Universities contribute to most of the P2P traffic traffic”. Many of these statistics can be predicted from the density of addresses assigned in the respective class C networks. ISC does share specific addresses in one place: it lists the top 10 scanners by IP address. Many organizations use this information to block misbehaving machines. The repercussion of doing this is again minimal. They do not provide specific details about those machines or the networks they are on. It simply serves to embarrass the ISPs that host the compromised machines. Any sort of anonymization here would defeat the purpose. Overall, the type of data they provide is a homogeneous set of aggregated statistics. More information can be gathered from the raw data it could be provided. Thus, one is not restricted to only the statistics they provide. The main difference of course is that the ISC is real-time and uses a more distributed sample. In conclusion, ISC works very well for monitoring general worm behavior, detecting trends that indicate new worms and analyzing the life cycle of an exploit. However, they are not gathering many types of logs, and they are not sharing them with the general community for research.



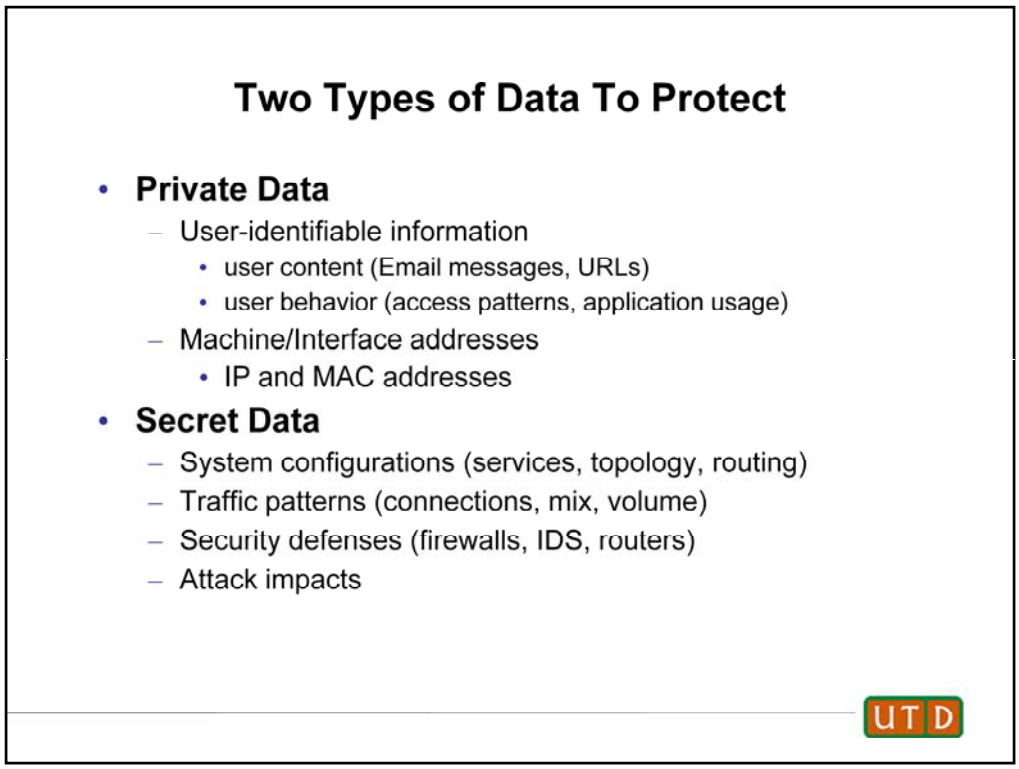
This figure shows examples of the state-of-the-art in security data sharing.

(1) This is the sharing we want, grass-roots balanced sharing between individual organizations. This is not occuring.

(2) This asymmetric centralized sharing – for example the centralized Internet Storm Center (ISC) sharing aggregate statistics from received logs. The amount of data received from distributed organizations is much greater than the amount of information made available in the form of aggregate statistics.

(3/4/5) represents asymmetric grass-roots sharing where direct sharing occurs between logically adjacent organizations (2<->3, 3<->4, 4<->5) but only indirect sharing between logically separated organizations (3<->5)

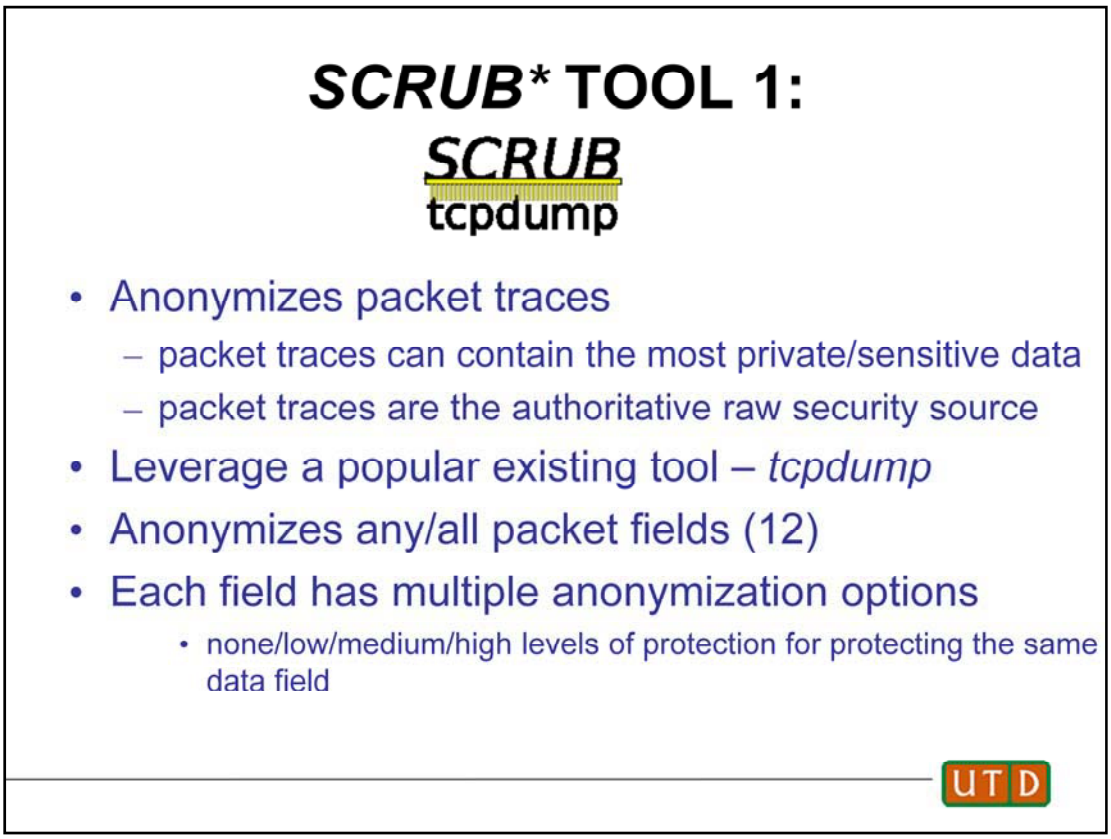
(6) An isolated organizations with no sharing between organization is all too common.



The practical concerns which have prevented organizations from sharing data include the resources needed to prepare data for sharing and a valid fear that private and/or sensitive information in shared data may be misused to cause harm.

Examples of private information in data that we may not want to reveal include personally-identifiable user information and user activities – often information that is protect by law.

Examples of secret information in data that we may not want to reveal includes system configurations, network topologies, network services, organizational defenses, and attack impacts – all valuable information to attackers.



As its name suggests, SCRUB-tcpdump builds upon the popular tcpdump tool for easy data management of packet traces while simultaneously protecting private/sensitive data from being disclosed through the use of anonymization. With SCRUB-tcpdump, a user can anonymize fields considered sensitive to multiple desired levels by selecting options that remove all information, add noise, or permute the data.

SCRUB-tcpdump use the libpcap engine to handle I/O, this enables input from either live capture or a capture log file. Our system reads, parses, and passes packets sequentially one packet at a time. Libpcap (as a stand-alone library) can be combined into a stand-alone SCRUB-tcpdump anonymizer or extended directly with extensions to tcpdump. We are in the process of extending tcpdump with patches and libraries for its released code. Functionality and output results will be the same using either stand-alone SCRUB-tcpdump or tcpdump extended with anonymization options – we have implemented the same command set in both. Since libpcap is the foundation upon which tcpdump rests, we find it well-suited for linking directly into tcpdump source, expanding command line options, and allowing users to utilize tcpdump directly for file handling as well as anonymizing during capture.

One of the major contributions of this work is that while other tools have focused primarily on IP address anonymization or only binary options for field anonymization, SCRUB-tcpdump allows a user to select multiple packet fields and each selected field has multiple anonymization options that may be applied specific to that field.

PACKET FIELDS [layer, bits]; (1) Fragmentation Flags [network, 3bits]; (2) IP Address [network, IPv4 32 bits]; (3) Payload [transport layer]; (4) Ports [transport layer, 16 bits] (5) Sequence Number [transport, 32 bits]; (6) TCP Flags [transport, 8 bits]; (7) Time Stamp [pcap]; (8) Time-To-Live (TTL) [network, 8 bits]; (9) Total Packet Length [network, 16 bits, and pcap]; (10) Transport Protocol Number Field [network, 8 bits]; (11) Type-Of-Service (TOS) [network; 8 bits]; (12) Window Size [transport layer; 16 bits].

*SCRUB** TOOL 2: *SCRUB-NetFlows*

- Anonymizes NetFlow logs
  - NetFlows logs efficiently aggregate packet traffic by connections
  - Most commonly shared security data
- Anonymizes any/all NetFlow fields (10)
- Each field has multiple anonymization options
  - none/low/medium/high levels of protection for protecting the same data field

UTD

A network flow (NetFlows) is defined as a sequence of IP packets that are transferred between two endpoints within a certain time

interval, with the most commonly used NetFlows formats are Cisco and Argus.

NetFlows Fields with multi-level options to anonymize

(1) IP address (source and destination)

(2/3) Timestamp (first and last packet)

(4) Ports Field (source and destination)

(5) Protocol Field

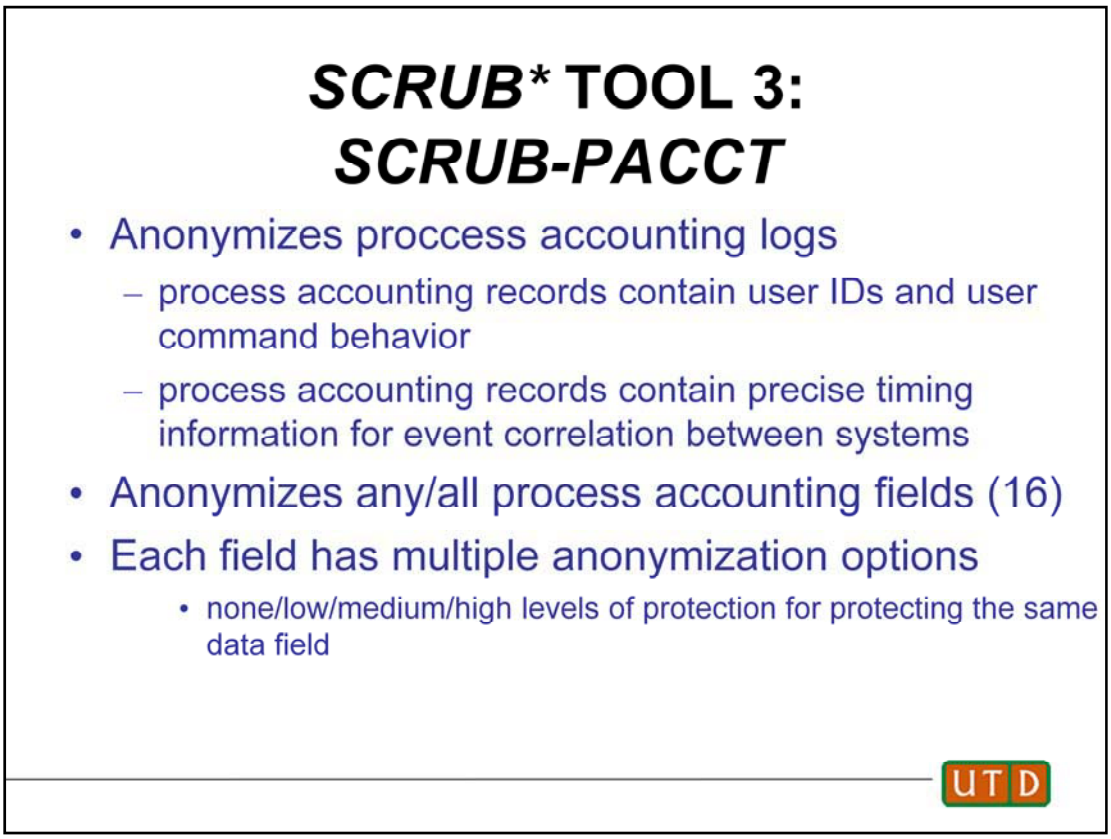
(6) Byte Count Field

(7) Type-of-Service (TOS)

(8) Time-to-Live (TTL)

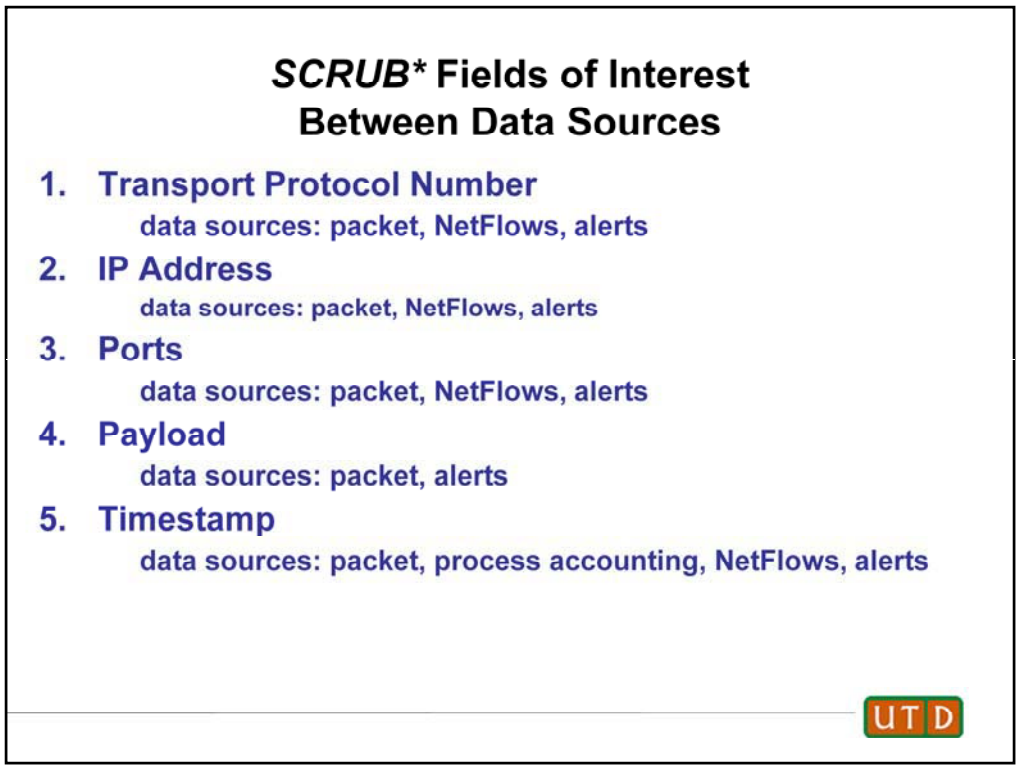
(9) Packet Count

(10) TCP Flags



The UNIX/Linux accounting system collects information on individual/group usage of computer system resources. A system can record every process created by every user. This logged data is called *process accounting* and has been found to be useful for several security purposes: Masquerade detection (illegitimate users); To hold a specific user accountable for some action indicated in the logs; To enable the extraction of patterns of use of objects, users, or security mechanisms in a system; To identify security policy violations; To create an audit trail of the use (or abuse) that may occur from a specific user

There are 16 fields within a process accounting record. *SCRUB-PACCT* has multiple anonymization options for each field.

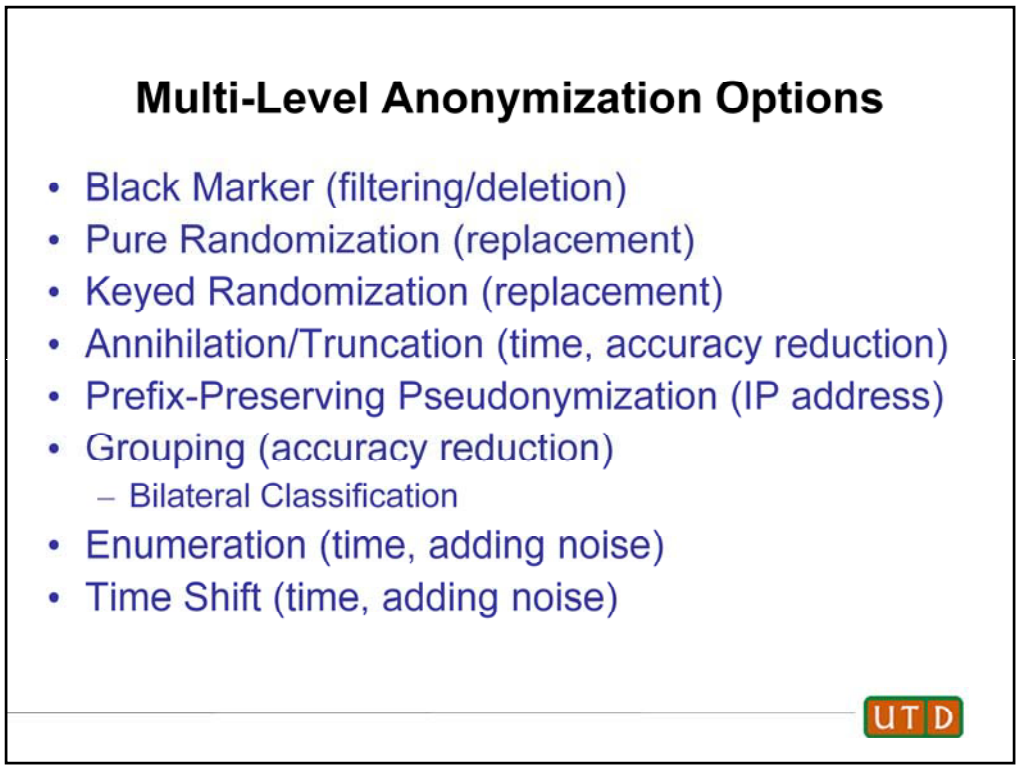


Multi-Field anonymization is necessary to prevent information leakage due to inference, however, many fields are in common between different data sources so this greatly decreases combinatorics.

In the *SCRUB** tool suite: *SCRUB-tcpdump, SCRUB-PACCT, SCRUB-NetFlows, and SCRUB-alerts (under-development) we process about 30 different fields.*

Of these thirty fields we find these five fields can be anonymized using the same set of algorithms and shared to identify/correlate events between organizations.

With these five fields as a basis upon which to build, *SCRUB** tools may provide an infrastructure for sharing of the data of choice at the appropriate level of protection (multi-level anonymization options).



Multi-Level Anonymization is necessary since there are no one-size-fits-all anonymization solutions – different options are needed. Different anonymization options provide different levels of privacy protection relative to the organizational security policy in question. Anonymized field values must still be valid field values so processing is invariant to anonymization. The anonymization options can be characterized as the following:

Filtering – deletion of field values

Replacement – pseudo-anonymous permutation mappings or fully-anonymous substitution mapping of field values

Reduction of Accuracy – approximating data values (examples include truncation, rounding) or grouping of field values

Adding Noise – adding noise to perturb field values (examples include time shifting)

Aggregation – summarization of field values with cumulative statistics

Here are examples of several specific anonymization options:

Black Marker All the information of the field is deleted by replacing every value of that field with a predefined constant matching the value type expected.

Pure Randomization Maps all input to a random value on output - however, all mappings of a given value X, which is randomly mapped to Y at its first encounter, will likewise be mapped to Y for the remainder of the file. Not reproducible.

Keyed Randomization A user-specified key and as the basis for a randomized permutation of the field. The randomization is reproduceable if the same key is reused.

Truncation Each original value replaced with a value truncated at a predefined truncation point. To truncate a value, we simply delete everything after the truncation point.

Prefix-Preserving Pseudonymization Separately anonymizes the subnet mask and the host portion of an IP address, using the Crypto-PAn algorithm thus ensuring that all hosts in the same subnet in the source file are also mapped to the same subnet in the resulting output.

Enumeration The enumeration method applies exclusively to timestamps. All records will be sorted sequentially in chronological order.

Grouping The complete set of values is partitioned and a canonical member of each equivalence class is chosen to replace all values from that equivalence class

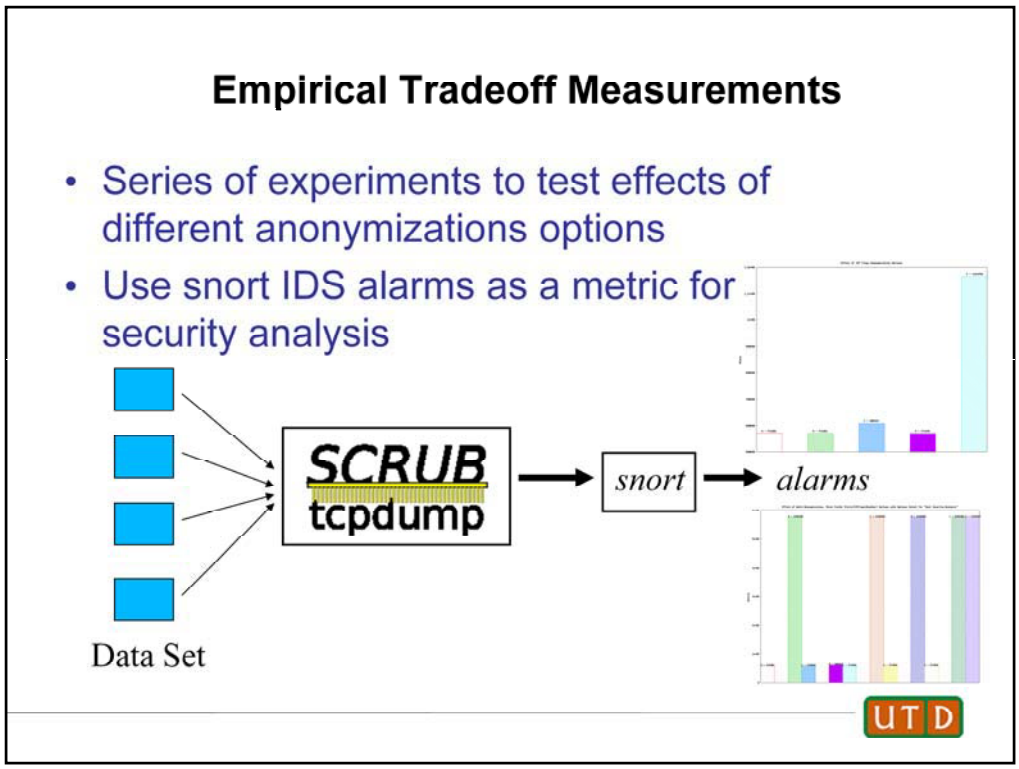
Privacy vs. Analysis Tradeoffs

while anonymization protects against information leakage it also destroys data needed for security analysis

- Zero-Sum? (more privacy <> less analysis & vice versa)
- to date, no quantitative measurements of how useful anonymized data is for security analysis

UTD

The fundamental tradeoff in data sharing between organizations using anonymization is the risk of valuable network data being unknowingly disclosed (privacy protection not stringent enough) versus valuable network data being needlessly deleted (security analysis maligned with privacy protection too stringent) – we refer to this as the privacy/analysis tradeoff. General intuition leads researchers to believe that anonymization is a zero sum tradeoff between privacy and analysis — the more network data is obscured for protection the less value it may be for security analysis. However, anonymization privacy/security tradeoffs have not been tested so no one really knows how real privacy/analysis tradeoffs behave.



The objective for experimentation is to quantitatively investigate the effect of *SCRUB** multi-level anonymization options on the tradeoff between privacy protection and security analysis in data to be shared. As shown in the above figure the experimental design is to compare measurements of a data set before and after executing different anonymization options. The metric we use as a proxy for security analysis is IDS alarms. We are aware, however, that IDS alarms are not a perfect proxy for security analysis. While less IDS alarms maps to lower levels of security analysis, the relationship of more IDS alarms to security analysis is non-linear. With more IDS alarms, more security analysis may have taken place if (and only if) new information is revealed by the new IDS alarms. However, more IDS alarms may also decrease ability to perform security analysis if the additional alarms are inaccurate or redundant. Despite this additional complexity, IDS alarms do provide a quantitative metric for security analysis and we carefully examine details about the nature of IDS alarms in the experimental results.

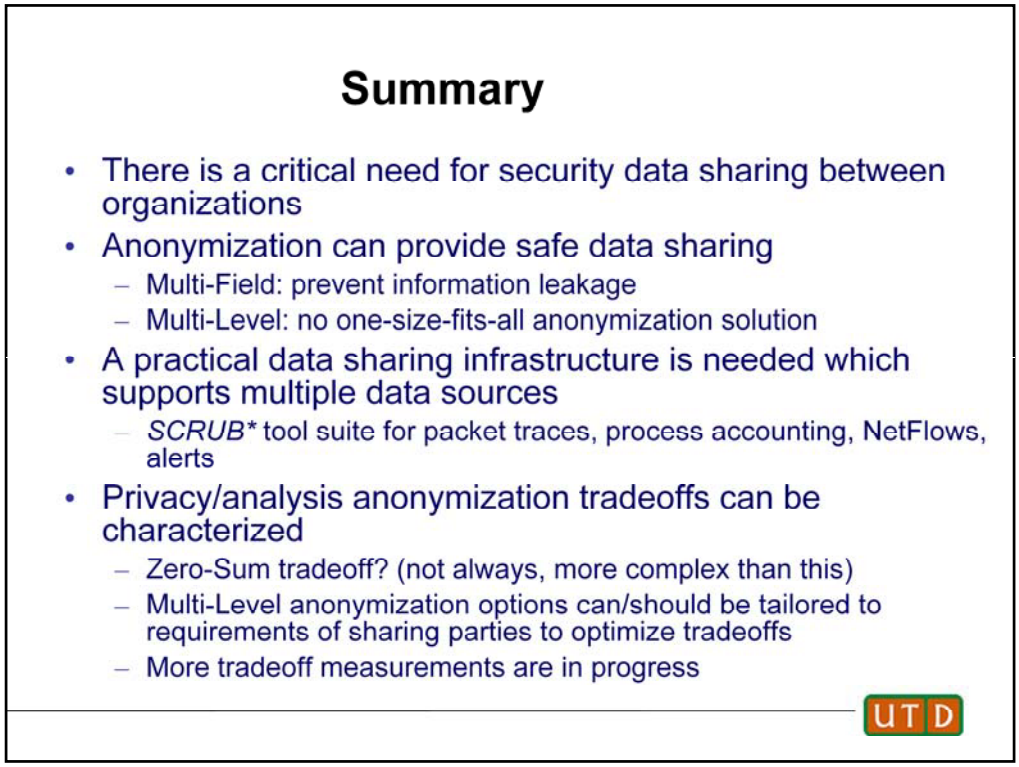
We select the Snort IDS for experimentation for two reasons: (1) we want a widely-used IDS so our experiments are reproducible as well as trusted, and (2) we require the ability to examine a standard open-source ruleset in order to understand why certain rules fire during experimentation. The ruleset utilized is the official "Sourcefire VRT Certified Rules" for version 2.24 with every rule turned on. This is the set of rules developed by the Sourcefire company that is continually kept up-to-date to include alerts for the newest and most critical security problems.

The two histograms have "number of alarms" on the vertical axis and different anonymization options on the horizontal axis.

The upper histogram figure presents preliminary results showing that different anonymization options on the same field provides different levels of protection and analysis.

The lower histogram figure presents preliminary results showing that different anonymization options on different combinations of fields provide different levels of protection and analysis.

Using this experimental design we find that not all privacy/analysis tradeoffs are zero sum. In fact, usage of network data anonymization tools for sharing creates complex tradeoffs that no one has yet studied. For example, port anonymization options for privacy have little impact on analysis so it is indeed possible simultaneously satisfy both privacy protection and security analysis requirements. We also have been able to show that selection of different multi-level anonymization options on the same data creates dramatically different privacy protection and security analysis capabilities. For example, the number of IDS alarms generated from anonymized network data can be about an order of magnitude different depending if the user selected anonymization options biased toward maximizing privacy protection or maximizing security analysis.



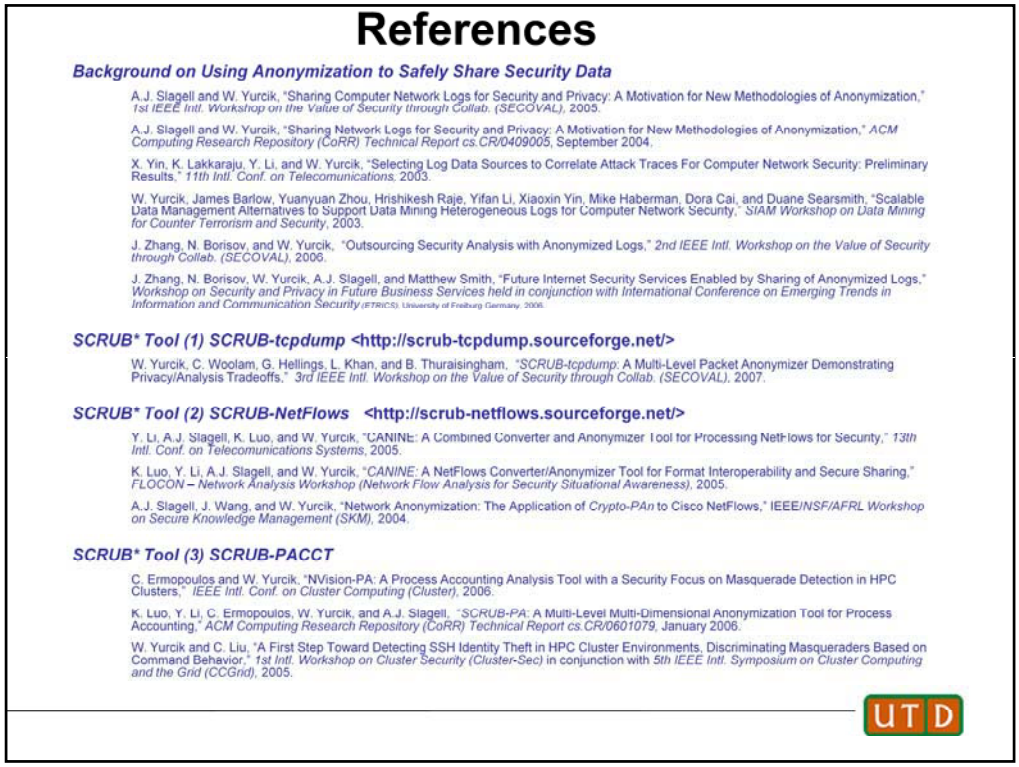
In summary, this work is a first step toward the safe sharing of data for distributed security analysis.

There is a critical need for security data sharing as the next level of protection against attackers, attackers are currently bouncing between targeted organizations due to lack of data sharing.

Anonymization can provide safe data sharing as long as multiple fields are anonymized to prevent data leakage from inference between fields, and as long as there are multiple anonymization options for each field since different organizations require different levels of protection.

At the practical level, an infrastructure is needed which supports multiple data sources since organizations need the freedom to select which data to share. The SCRUB* suite of tools provides this infrastructure based on common algorithms for common fields between data sources.

However, not until anonymization privacy/analysis tradeoffs are well understood will trusted sharing using anonymization occur. While anonymization tradeoffs have been speculated upon by many researchers, more work is needed to measure these tradeoffs. We already know from preliminary results that these are complex tradeoffs, not simple zero sum tradeoffs. When these tradeoffs are characterized then organizations can select how they want to optimize their security data sharing given privacy constraints and need for security analysis.



References to previous work by the authors related to the *SCRUB** architecture.

General Motivation papers for safe sharing

And then papers broken out for three of the *SCRUB** tools. The *SCRUB-alerts* tool is still in development and has no references as yet.