

# Anonymization of SNMP traces

Matúš Harvan

March 29, 2005

# What is Simple Network Management Protocol (SNMP)

SNMP:

- protocol to access management and control information of network devices
- lightweight
- used extensively in enterprise networks and by ISPs, especially for monitoring purposes

# Why do we need SNMP traces?

- it is believed that SNMP is used differently in different environments and various SNMP implementations perform differently
- due to lack of available traces from operational networks, not known how exactly SNMP is used in practice, which particular management applications are preferred and how efficiently they make use of available protocol options
- need to *anonymize* in such a way that:
  - privacy and security of originating networks is not breached
  - traces would retain enough information to be useful for SNMP analysis

# Why do we need SNMP traces?

- it is believed that SNMP is used differently in different environments and various SNMP implementations perform differently
- due to lack of available traces from operational networks, not known how exactly SNMP is used in practice, which particular management applications are preferred and how efficiently they make use of available protocol options
- need to *anonymize* in such a way that:
  - privacy and security of originating networks is not breached
  - traces would retain enough information to be useful for SNMP analysis

# Why do we need SNMP traces?

- it is believed that SNMP is used differently in different environments and various SNMP implementations perform differently
- due to lack of available traces from operational networks, not known how exactly SNMP is used in practice, which particular management applications are preferred and how efficiently they make use of available protocol options
- need to *anonymize* in such a way that:
  - privacy and security of originating networks is not breached
  - traces would retain enough information to be useful for SNMP analysis

# Subproblems

Two particular challenges:

- prefix-preserving and lexicographical-order-preserving IP address anonymization
  - prefix relationships might be important
  - tables stored in lexicographical order and retrieved sequentially
- anonymization of complete SNMP traces (packets including payload)

# Subproblems

Two particular challenges:

- prefix-preserving and lexicographical-order-preserving IP address anonymization
  - prefix relationships might be important
  - tables stored in lexicographical order and retrieved sequentially
- anonymization of complete SNMP traces (packets including payload)

# Prefix-preserving IP address anonymization

prefix-preserving anonymization solved:

- *tcpdpriv*
  - not consistent, not parallel
  - security problems (-A50 option)
  - operates only on IP, TCP and UDP headers – scrambles payload
- *Crypto-PAn*
  - canonical form for all prefix-preserving anonymization functions
  - using cryptography (Rijndael) for anonymization
  - operates only on IP addresses

# Prefix-preserving IP address anonymization

prefix-preserving anonymization solved:

- *tcpdpriv*
  - not consistent, not parallel
  - security problems (-A50 option)
  - operates only on IP, TCP and UDP headers – scrambles payload
- *Crypto-PAn*
  - canonical form for all prefix-preserving anonymization functions
  - using cryptography (Rijndael) for anonymization
  - operates only on IP addresses

## Prefix-preserving Anonymization[3]

Two IP addresses  $a = a_1 a_2 \dots a_n$  and  $b = b_1 b_2 \dots b_n$  share a  $k$ -bit prefix ( $0 \leq k \leq n$ ) if  $a_1 a_2 \dots a_k = b_1 b_2 \dots b_k$  and  $a_{k+1} \neq b_{k+1}$  when  $k < n$ . An anonymization function  $F$  is defined as one-to-one function from  $\{0, 1\}^n$  to  $\{0, 1\}^n$ . An anonymization function  $F$  is prefix-preserving if given two IP addresses  $a$  and  $b$  that share a  $k$ -bit prefix,  $F(a)$  and  $F(b)$  share a  $k$ -bit prefix as well.

# Canonical Form Theorem[3]

Let  $f_i$  be a function from  $\{0, 1\}^i$  to  $\{0, 1\}$  for  $i = 1, 2, \dots, n - 1$  and  $f_0$  be a constant function. Let  $F$  be a function from  $\{0, 1\}^n$  to  $\{0, 1\}^n$  defined as follows. Given  $a = a_1 a_2 \dots a_n$ , let

$$F(a) := a'_1 a'_2 \dots a'_n$$

where  $a'_i = a_i \oplus f_{i-1}(a_1, a_2, \dots, a_{i-1})$  and  $\oplus$  is the exclusive-or operation, for  $i = 1, 2, \dots, n$ . Then  $F$  is a prefix-preserving anonymization function and every prefix-preserving anonymization function necessarily takes this form.

# Address Tree

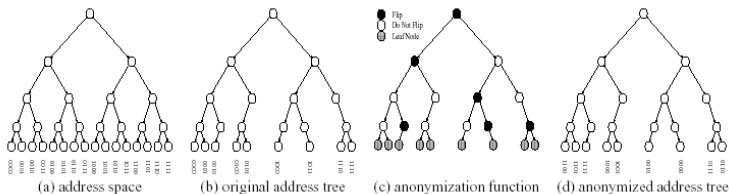


figure taken from [3]

## Lexicographical order on IP addresses

Let  $a = o_1^a o_2^a \dots o_n^a$  and  $b = o_1^b o_2^b \dots o_n^b$  be two IP addresses (of the same length) where  $o$ 's are octets and  $<^l$  be a lexicographic ordering. Then

$$a <^l b \Leftrightarrow o_1^a o_2^a \dots o_n^a <^l o_1^b o_2^b \dots o_n^b$$
$$\Leftrightarrow (\exists m > 0)(\forall i < m)(o_i^a = o_i^b) \wedge (o_m^a < o_m^b)$$

[3]

e.g. 1.2.3.4  $<^l$  1.12.3.4

An anonymization function  $F$  is a one-to-one function from  $\{0, 1\}^n$  to  $\{0, 1\}^n$ .  $F$  is lexicographical-order-preserving if given two IP addresses  $a$  and  $b$  we have:

$$a <^l b \Rightarrow F(a) <^l F(b)$$

*used<sub>i</sub>*

Let  $used_i$  be a function from  $\{0, 1\}^i$  to  $\{0, 1\}$  for  $i = 0, 1, 2, \dots, n - 1$ . This function determines if the IP addresses in the subtree below the  $a_i$  bit are used.  $used_i$  is defined recursively as

$$used_{i-1}(a_1 a_2 \dots a_{i-1}) = used_i(a_1 a_2 \dots a_{i-1} 0) \vee used_i(a_1 a_2 \dots a_{i-1} 1)$$

and  $used_n(a_1 a_2 \dots a_i)$  is true if the IP address  $a_1 a_2 \dots a_i$  is in the traffic trace and false otherwise. Then

$$used_0 = used_1(0) \vee used_1(1).$$

*canflip<sub>i</sub>*

Let *canflip<sub>i</sub>* be a functions from  $\{0, 1\}^i$  to  $\{0, 1\}$  for  $i = 0, 1, 2, \dots, n - 1$ . This function determines if the bit  $a_{i+1}$  of an IP address  $a_1 a_2 \dots a_n$  can be flipped while still preserving the lexicographical order. It is defined as

$$\begin{aligned} & \text{canflip}_{i-1}(a_1 a_2 \dots a_{i-1}) = \\ & \neg (\text{used}_i(a_1 a_2 \dots a_{i-1} 0) \wedge \text{used}_i(a_1 a_2 \dots a_{i-1} 1)) \end{aligned}$$

# Lexicographical-order-preserving Anonymization

Let  $f_i$  be a function from  $\{0, 1\}^n$  to  $\{0, 1\}$  for  $i = 1, 2, \dots, n - 1$  and  $f_0$  be a constant function. Let  $F$  be a function from  $\{0, 1\}^n$  to  $\{0, 1\}^n$  defined as follows. Given  $a = a_1 a_2 \dots a_n$ , let

$$F(a) := a'_1 a'_2 \dots a'_n$$

where

$$a'_i = a_i \oplus (f_{i-1}(a_1, a_2, \dots, a_{i-1}) \wedge \text{canflip}_{i-1}(a_1, a_2, \dots, a_{i-1}))$$

for  $i = 1, 2, \dots, n$ . Then we claim  $F$  is a prefix-preserving and lexicographical-order-preserving anonymization function.

## Related projects doing complete anonymization

- Projects:
  - anonymization of ftp traces
  - router configuration anonymization
- both projects follow the *filter-in principle* – only known information allowed to pass through unmodified, unknown gets scrambled
- allows to verify correctness and increases trust of network owners

## Related projects doing complete anonymization

- Projects:
  - anonymization of ftp traces
  - router configuration anonymization
- both projects follow the *filter-in principle* – only known information allowed to pass through unmodified, unknown gets scrambled
- allows to verify correctness and increases trust of network owners

## Related projects doing complete anonymization

- Projects:
  - anonymization of ftp traces
  - router configuration anonymization
- both projects follow the *filter-in principle* – only known information allowed to pass through unmodified, unknown gets scrambled
- allows to verify correctness and increases trust of network owners

## Related projects doing complete anonymization

- Projects:
  - anonymization of ftp traces
  - router configuration anonymization
- both projects follow the *filter-in principle* – only known information allowed to pass through unmodified, unknown gets scrambled
- allows to verify correctness and increases trust of network owners




## Goals of this guided research project:

- find a suitable IP address anonymization scheme
- develop a tool for anonymization of complete SNMP traffic traces

## References:

-  Greg Minshall.  
tcpdpriv, 1996.  
<http://ita.ee.lbl.gov/html/contrib/tcpdpriv.html>.
-  Jun Xu, Jinliang Fan, Mostafa H. Ammar, and Sue Moon.  
Crypto-pan, 2003.  
<http://www.cc.gatech.edu/computing/Telecomm/cryptopan/>.
-  Jun Xu, Jinliang Fan, and Mostafa H. Ammar.  
Prefix-preserving IP address anonymization:  
measurement-based security evaluation and a new  
cryptography-based scheme.  
*In Proceedings of the 10 th IEEE International Conference on  
Network Protocols (ICNP'02), 2002.*

## References:

-  David A. Maltz, Jibin Zhan, Geoffrey Xie, and Hui Zhang. Structure preserving anonymization of router configuration data.  
*In Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, 2004.*
-  Rouming Pang and Vern Paxson.  
A high-level programming environment for packet trace anonymization and transformation.  
*In Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications, 2003.*
-  [http://encyclopedia.laborlawtalk.com/Lexicographic\\_order](http://encyclopedia.laborlawtalk.com/Lexicographic_order).