Traffic analyzers are extremely valuable tools for traffic control, protocol analysis, anomaly identification, monitoring, etc. When using these invasive technologies, users’ security and privacy must be considered. This can be accomplished by modifying specific header parameters to protect individuals. Our approach is to use Cryptography-Based and Prefix-Preserving Anonymization algorithm in tcpdump to grant anonymity to the users.

### Introduction

- Traffic analyzers are extremely valuable tools for traffic control, protocol analysis, anomaly identification, monitoring, etc.
- When using these invasive technologies, users’ security and privacy must be considered.
- This can be accomplished by modifying specific header parameters to protect individuals.
- Our approach is to use Cryptography-Based and Prefix-Preserving Anonymization algorithm in tcpdump to grant anonymity to the users.

### Prefix-Preserving Anonymization

- For any pair of strings $x$ and $y$ that share a common prefix of length $p$.
- Counterparts $E_k(x), E_k(y)$ will share a common prefix of length $p$.

### Motivation

1. When a packet can be uniquely associated with a specific user, such as an internal network worker or a client, issues of privacy arise.
2. Prefix-Preserving anonymization allows us to keep the subnet structure of IP addresses while also providing users with the necessary privacy.

### Approach

- Set a **new flag** in tcpdump to anonymize the packet using cryptopANT library before dumping, storing or displaying it.
- An **algorithm** (such as Blowfish, AES, or MD5) and a **key** can be specified to encrypt the IP addresses. If no options are specified, default settings are applied and a key is produced from /dev/urandom.
- After the initialization of cryptopANT, the handler, that would normally be executed every time a packet is intercepted, will be saved and invoked after the anonymization function.
- The **anonymization function** will process the packet by searching for IP addresses in the Network and Application layer headers and anonymizing them using the cryptopANT library’s encryption capabilities.
- As shown in Figure 1, after anonymization, the callback reference saved at the beginning of the process will be called.
- Some of the **protocols** that will be supported by this extension are the following: IPv4, IPv6, ICMP, ICMPv6, ARP, RARP, NDP, DNS, DHCP and IP in IP.
- In order to **undo the anonymization**, we would introduce a new flag. If the keys for anonymize and deanonymize are the same, this operation will return the right IP address.

### Results

- Figure 2 presents a demo of IP anonymization of $p = 16$.
- Using AES-128-ECB the anonymization process takes about $\approx 0.01\text{ms per packet}$.

<table>
<thead>
<tr>
<th>$IP_x = 10.0.1.15$</th>
<th>$IP_y = 10.0.2.15$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E_k(IP_x)$</td>
<td>$E_k(IP_y)$</td>
</tr>
<tr>
<td>$IP_x = 82.223.65.115$</td>
<td>$IP_y = 82.223.67.76$</td>
</tr>
</tbody>
</table>

### Future Work

- Pull request to the tcpdump repository to add this operation.
- Cryptographic evaluation of the anonymization function and cryptopANT library to determine the possibility of attacks.
- Definition of additional functions (other than the anon function) that modify the packet before dumping or displaying it, such as eliminating sensible pieces of headers.