Network Log Anonymization: Application of Crypto-PAn to Cisco Netflows

Adam Slagell, Jun Wang and William Yurcik,

National Center for Supercomputing Applications (NCSA)
University of Illinois at Urbana-Champaign
Motivation for Sharing Logs

• **Share for**
  – *Security Research*
    • *Create better detection tools and test them*
  – *Security Operations*
  – *Network Measurements*

• **Who says it's important?**
  – *DHS with Information Sharing and Analysis Centers*
  – *National Strategy to Secure Cyberspace*

• **Why Netflows?**
IP Anonymization Techniques

• Black Marker Effect
  – Great information loss
  – Cannot correlate attacks against machine X

• Truncation
  – Finer grained control of information loss
  – Used for Source Report at ISC
    • Origins of scans

• Random Permutations
  – Injective Mapping, a type of pseudonymization
  – Allows correlation but destroys structure
Prefix-Preserving Anonymization

• Definition
  – Let P be a permutation of the set of IP addresses
  – P is a prefix-preserving anonymization function if and only if for all IP addresses x and y:
    • x and y match on exactly the same length prefix as P(x) and P(y)

• Preserves subnet structures and relationships

• Structure can of course be exploited by attackers
Prefix Preserving Tools

- **Crypto-PAn**
  - Key based solution
- **TCPdpriv**
  - Table based solution for TCPdump files
- **Ip2anonip**
  - A filter to anonymize IP addresses based off TCPdpriv
- **Ipsumdump**
  - Summarizes TCP/IP dumps
  - Optionally performs prefix-preserving anonymization based off TCPdpriv
What We Have Done

• The problem:
  – Our visualization tools use netflows
  – We need students to work on these projects
  – Information is sensitive

• Subnet structure is vital to tools. Thus Crypto-PAn is ideal.

• No key generator in Crypto-Pan

• Created a pass-phrase based key generator without extra libraries
Key Generator

- Input passphrase (unechoed), max 256 bytes
- Wrap till buffer filled
- CBC encrypt with fixed key
  - This combines data to create an intermediate key
    - Why can’t we just XOR blocks?
  - Cannot stop here, processes is reversible
- Use the intermediate key to re-encrypt the original buffer
  - Take the last 32 bytes as the end key
    - Even without dropping 244 bytes, this is irreversible
Performance

• Work on binary logs
  • Avoids extra conversions

• On laptop still less than 20 minutes for 2 Gigabytes of flows

<table>
<thead>
<tr>
<th>MACHINE (GHz)</th>
<th>Records/Second</th>
<th>Total Time (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dual 2.4 Xeon</td>
<td>75015.342</td>
<td>10.45</td>
</tr>
<tr>
<td>Single 2.4 Xeon</td>
<td>42686.279</td>
<td>18.37</td>
</tr>
<tr>
<td>1.7 Pentium M</td>
<td>40113.674</td>
<td>19.55</td>
</tr>
</tbody>
</table>
Conclusions & Future Work

- Feasible solution for even large universities
  - Provides high utility, but lower security
- Many attacks on anonymization schemes
  - Inference attacks, chosen plaintext, structure exploitation
- Need new options to balance utility & security
  - Different levels of anonymization
    - Means considering more fields
  - Different types of logs
Thank You

• **Email:** slagell@ncsa.uiuc.edu

• **Links of Interest**