Network Security

Tadayoshi Kohno

Some slides based on Vitaly Shmatikov’s
Detecting Attack Strings Is Hard

- Want to detect “USER root” in packet stream
- Scanning for it in every packet is not enough
  - Attacker can split attack string into several packets; this will defeat stateless NIDS
- Recording previous packet’s text is not enough
  - Attacker can send packets out of order
- Full reassembly of TCP state is not enough
  - Attacker can use TCP tricks so that certain packets are seen by NIDS but dropped by the receiving application
    - Manipulate checksums, TTL (time-to-live), fragmentation
TCP Attacks on NIDS

Insertion attack

Insert packet with bogus checksum

TTL attack

Short TTL to ensure this packet doesn’t reach destination

Dropped (TTL expired)
Anomaly Detection with NIDS

- **Advantage:** can recognize new attacks and new versions of old attacks

- **Disadvantages**
  - High false positive rate
  - Must be trained on known good data
    - Training is hard because network traffic is very diverse
  - Protocols are finite-state machines, but current state of a connection is difficult to see from the network
  - Definition of “normal” constantly evolves
    - What’s the difference between a flash crowd and a denial of service attack?
Intrusion Detection Problems

- Lack of training data with real attacks
  - But lots of “normal” network traffic, system call data

- Data drift
  - Statistical methods detect changes in behavior
  - Attacker can attack gradually and incrementally

- Main characteristics not well understood
  - By many measures, attack may be within bounds of “normal” range of activities

- False identifications are very costly
  - Sysadm will spend many hours examining evidence
Intrusion Detection Errors

- **False negatives:** attack is not detected
  - Big problem in signature-based misuse detection
- **False positives:** harmless behavior is classified as an attack
  - Big problem in statistical anomaly detection
- Both types of IDS suffer from both error types
- Which is a bigger problem?
  - Attacks are fairly rare events
Conditional Probability

- Suppose two events A and B occur with probability $\Pr(A)$ and $\Pr(B)$, respectively.
- Let $\Pr(AB)$ be probability that both A and B occur.
- What is the conditional probability that A occurs assuming B has occurred?
Conditional Probability

- Suppose two events A and B occur with probability $Pr(A)$ and $Pr(B)$, respectively.
- Let $Pr(AB)$ be probability that both A and B occur.
- What is the conditional probability that A occurs assuming B has occurred?

$$Pr(A \mid B) = \frac{Pr(AB)}{Pr(B)}$$
Bayes’ Theorem

- Suppose mutually exclusive events $E_1, \ldots, E_n$ together cover the entire set of possibilities.
- Then probability of any event $A$ occurring is
  \[ \Pr(A) = \sum_{1 \leq i \leq n} \Pr(A \mid E_i) \cdot \Pr(E_i) \]
  - Intuition: since $E_1, \ldots, E_n$ cover entire probability space, whenever $A$ occurs, some event $E_i$ must have occurred.

- Can rewrite this formula as
  \[ \Pr(E_i \mid A) = \frac{\Pr(A \mid E_i) \cdot \Pr(E_i)}{\Pr(A)} \]
1% of traffic is SYN floods; IDS accuracy is 90%
- IDS classifies a SYN flood as attack with prob. 90%,
  classifies a valid connection as attack with prob. 10%

- What is the probability that a connection flagged
  by IDS as a SYN flood is actually valid traffic?
1% of traffic is SYN floods; IDS accuracy is 90% 
- IDS classifies a SYN flood as attack with prob. 90%, classifies a valid connection as attack with prob. 10%

What is the probability that a connection flagged by IDS as a SYN flood is actually valid traffic?

\[
Pr(\text{valid} | \text{alarm}) = \frac{Pr(\text{alarm} | \text{valid}) \cdot Pr(\text{valid})}{Pr(\text{alarm})}
\]
1% of traffic is SYN floods; IDS accuracy is 90%.

- IDS classifies a SYN flood as attack with prob. 90%,
  classifies a valid connection as attack with prob. 10%

What is the probability that a connection flagged by IDS as a SYN flood is actually valid traffic?

$$\Pr(\text{valid} | \text{alarm}) = \frac{\Pr(\text{alarm} | \text{valid}) \cdot \Pr(\text{valid})}{\Pr(\text{alarm})}$$

$$\Pr(\text{valid} | \text{alarm}) = \frac{\Pr(\text{alarm} | \text{valid}) \cdot \Pr(\text{valid})}{\Pr(\text{alarm} | \text{valid}) \cdot \Pr(\text{valid}) + \Pr(\text{alarm} | \text{SYN flood}) \cdot \Pr(\text{SYN flood})}$$
1% of traffic is SYN floods; IDS accuracy is 90% 
• IDS classifies a SYN flood as attack with prob. 90%, 
  classifies a valid connection as attack with prob. 10%

What is the probability that a connection flagged by IDS as a SYN flood is actually valid traffic?

$$\Pr(\text{valid} | \text{alarm}) = \frac{\Pr(\text{alarm} | \text{valid}) \cdot \Pr(\text{valid})}{\Pr(\text{alarm}) \cdot \Pr(\text{alarm} | \text{valid}) \cdot \Pr(\text{valid})} = \frac{\Pr(\text{alarm} | \text{valid}) \cdot \Pr(\text{valid}) + \Pr(\text{alarm} | \text{SYN flood}) \cdot \Pr(\text{SYN flood})}{0.10 \cdot 0.99} = \frac{0.10 \cdot 0.99 + 0.90 \cdot 0.01}{0.10 \cdot 0.99}$$
Base-Rate Fallacy

- 1% of traffic is SYN floods; IDS accuracy is 90%
  - IDS classifies a SYN flood as attack with prob. 90%, classifies a valid connection as attack with prob. 10%
- What is the probability that a connection flagged by IDS as a SYN flood is actually valid traffic?

Pr(valid | alarm) = \frac{Pr(alarm | valid) \cdot Pr(valid)}{Pr(alarm)}

= \frac{Pr(alarm | valid) \cdot Pr(valid) + Pr(alarm | SYN flood) \cdot Pr(SYN flood)}{Pr(alarm | valid) \cdot Pr(valid) + Pr(alarm | SYN flood) \cdot Pr(SYN flood)}

= \frac{0.10 \cdot 0.99 + 0.90 \cdot 0.01}{0.10 \cdot 0.99 + 0.90 \cdot 0.01} = 92% chance raised alarm is false!!!
Network Telescopes and Honeypots

- Monitor a cross-section of Internet address space
  - Especially useful if includes unused “dark space”

- Attacks in far corners of the Internet may produce traffic directed at your addresses
  - “Backscatter”: responses of DoS victims to randomly spoofed IP addresses
  - Random scanning by worms

- Can combine with “honeypots”
  - Any outbound connection from a “honeypot” behind an otherwise unused IP address means infection (why?)
  - Can use this to extract worm signatures (how?)
Anonymity
Privacy on Public Networks

- Internet is designed as a public network
  - Machines on your LAN may see your traffic, network routers see all traffic that passes through them

- Routing information is public
  - IP packet headers identify source and destination
  - Even a passive observer can easily figure out who is talking to whom

- Encryption does not hide identities
  - Encryption hides payload, but not routing information
  - Even IP-level encryption (tunnel-mode IPSec/ESP) reveals IP addresses of IPSec gateways
Applications of Anonymity (I)

❖ Privacy
  • Hide online transactions, Web browsing, etc. from intrusive governments, marketers and archivists

❖ Untraceable electronic mail
  • Corporate whistle-blowers
  • Political dissidents
  • Socially sensitive communications (online AA meeting)
  • Confidential business negotiations

❖ Law enforcement and intelligence
  • Sting operations and honeypots
  • Secret communications on a public network
Applications of Anonymity (II)

- Digital cash
  - Electronic currency with properties of paper money (online purchases unlinkable to buyer’s identity)
- Anonymous electronic voting
- Censorship-resistant publishing
What is Anonymity?

- Anonymity is the state of being not identifiable within a set of subjects
  - You cannot be anonymous by yourself!
    - Big difference between anonymity and confidentiality
  - Hide your activities among others’ similar activities
- Unlinkability of action and identity
  - For example, sender and his email are no more related after observing communication than they were before
- Unobservability (hard to achieve)
  - Any item of interest (message, event, action) is indistinguishable from any other item of interest
Attacks on Anonymity

- Passive traffic analysis
  - Infer from network traffic who is talking to whom
  - To hide your traffic, must carry other people’s traffic!

- Active traffic analysis
  - Inject packets or put a timing signature on packet flow

- Compromise of network nodes
  - Attacker may compromise some routers
  - It is not obvious which nodes have been compromised
    - Attacker may be passively logging traffic
  - Better not to trust any individual router
    - Assume that some fraction of routers is good, don’t know which
Chaum’s Mix

- Early proposal for anonymous email

- Public key crypto + trusted re-mailer (Mix)
  - Untrusted communication medium
  - Public keys used as persistent pseudonyms

- Modern anonymity systems use Mix as the basic building block
Chaum’s Mix

◆ Early proposal for anonymous email

◆ Public key crypto + trusted re-mailer (Mix)
  • Untrusted communication medium
  • Public keys used as persistent pseudonyms

◆ Modern anonymity systems use Mix as the basic building block

Before spam, people thought anonymous email was a good idea 😊
Basic Mix Design
Basic Mix Design

\{r_1, r_0, M\}_{pk(B), B}_{pk(mix)}
Basic Mix Design

\[{r_1, r_0, M}_{pk(B), B}_{pk(mix)} \}

\[{r_0, M}_{pk(B), B} \]
Basic Mix Design

\[\{r_1, \{r_0, M\}_{pk(B)}, B\}_{pk(mix)}\]

\[\{r_2, \{r_3, M'\}_{pk(E)}, E\}_{pk(mix)}\]

\[\{r_0, M\}_{pk(B)}, B\]
Basic Mix Design

A \{r_1, \{r_0, M\}_{\text{pk(B)}}, B\}_{\text{pk(mix)}}

C \{r_2, \{r_3, M'\}_{\text{pk(E)}}, E\}_{\text{pk(mix)}}

D \{r_4, \{r_5, M''\}_{\text{pk(B)}}, B\}_{\text{pk(mix)}}

Mix

B \{r_0, M\}_{\text{pk(B)}}, B

E
Basic Mix Design

\{r_1, \{r_0, M\}_{pk(B)}, B\}_{pk(mix)}

\{r_2, \{r_3, M'\}_{pk(E)}, E\}_{pk(mix)}

\{r_3, \{r_5, M''\}_{pk(B)}, B\}_{pk(mix)}

\{r_4, \{r_5, M''\}_{pk(B)}, B\}_{pk(mix)}

\{r_0, M\}_{pk(B)}, B

\{r_5, M''\}_{pk(B), B}

\{r_3, M'\}_{pk(E), E}
Basic Mix Design

Adversary knows all senders and all receivers, but cannot link a sent message with a received message.
Anonymous Return Addresses

\[ \{r_1, \{r_0, M\}_\text{pk(B)}, B\}_\text{pk(mix)} \]

\[ \{r_0, M\}_\text{pk(B)}, B \]
Anonymous Return Addresses

M includes \(\{K_1, A\}_{pk(mix)}\), \(K_2\) where \(K_2\) is a fresh public key

\[
\{r_1, \{r_0, M\}_{pk(B)}, B\}_{pk(mix)}
\]

\[
\{r_0, M\}_{pk(B)}, B
\]
Anonymous Return Addresses

M includes $\{K_1, A\}_{pk(mix)}$, $K_2$ where $K_2$ is a fresh public key

$\{r_1, \{r_0, M\}_{pk(B), B}\}_{pk(mix)}$

Response MIX

$\{K_1, A\}_{pk(mix)}$, $\{r_2, M'\}_{K_2}$

$\{r_0, M\}_{pk(B), B}$
Anonymous Return Addresses

M includes \( \{K_1, A\}_{pk(mix)} \), \( K_2 \) where \( K_2 \) is a fresh public key

\[ \{r_1, \{r_0, M\}_{pk(B)}, B\}_{pk(mix)} \]

\[ \{r_0, M\}_{pk(B)}, B \]

\[ A, \{\{r_2, M'\}_{K_2}\}_{K_1} \]

\[ \{K_1, A\}_{pk(mix)}, \{r_2, M'\}_{K_2} \]
Mix Cascade

- Messages are sent through a sequence of mixes
  - Can also form an arbitrary network of mixes ("mixnet")
- Some of the mixes may be controlled by attacker, but even a single good mix guarantees anonymity
- Pad and buffer traffic to foil correlation attacks
Disadvantages of Basic Mixnets

- Public-key encryption and decryption at each mix are computationally expensive
- Basic mixnets have high latency
  - Ok for email, not Ok for anonymous Web browsing
- Challenge: low-latency anonymity network
  - Use public-key cryptography to establish a “circuit” with pairwise symmetric keys between hops on the circuit
  - Then use symmetric decryption and re-encryption to move data messages along the established circuits
  - Each node behaves like a mix; anonymity is preserved even if some nodes are compromised
Another Idea: Randomized Routing

- Hide message source by routing it randomly
  - Popular technique: Crowds, Freenet, Onion routing
- Routers don’t know for sure if the apparent source of a message is the true sender or another router
Onion Routing

[Reed, Syverson, Goldschlag ’97]

Sender chooses a random sequence of routers

- Some routers are honest, some controlled by attacker
- Sender controls the length of the path
Route Establishment

- Routing info for each link encrypted with router’s public key
- Each router learns only the identity of the next router
Route Establishment

- Routing info for each link encrypted with router’s public key
- Each router learns only the identity of the next router
Route Establishment

- Routing info for each link encrypted with router’s public key
- Each router learns only the identity of the next router
Route Establishment

- Routing info for each link encrypted with router’s public key
- Each router learns only the identity of the next router
Route Establishment

- Routing info for each link encrypted with router’s public key
- Each router learns only the identity of the next router
Route Establishment

- Routing info for each link encrypted with router’s public key
- Each router learns only the identity of the next router

\[ \{M\}_{pk(B)} \]
Tor

- Second-generation onion routing network
  - [http://tor.eff.org](http://tor.eff.org)
  - Developed by Roger Dingledine, Nick Mathewson and Paul Syverson
  - Specifically designed for low-latency anonymous Internet communications
- Running since October 2003
- 100 nodes on four continents, thousands of users
- “Easy-to-use” client proxy
  - Freely available, can use it for anonymous browsing
Tor Management Issues

- Many applications can share one circuit
  - Multiple TCP streams over one anonymous connection
- Tor router doesn’t need root privileges
  - Encourages people to set up their own routers
  - More participants = better anonymity for everyone
- Directory servers
  - Maintain lists of active onion routers, their locations, current public keys, etc.
  - Control how new routers join the network
    - “Sybil attack”: attacker creates a large number of routers
  - Directory servers’ keys ship with Tor code
Deployed Anonymity Systems

- Free Haven project has an excellent bibliography on anonymity
  - http://freehaven.net
- Tor (http://tor.eff.org)
  - Overlay circuit-based anonymity network
  - Best for low-latency applications such as anonymous Web browsing
- Mixminion (http://www.mixminion.net)
  - Network of mixes
  - Best for high-latency applications such as anonymous email
FoxTor, Images from http://cups.cs.cmu.edu/foxtor/
FoxTor, Images from http://cups.cs.cmu.edu/foxtor/
FoxTor, Images from http://cups.cs.cmu.edu/foxtor/
Information Leakage
Stepping Stones
(courtesy of Yin Zhang)

◆ IP traceback helps discover machines from which attack packets originate
  • These often have remote-controlled zombie daemons
  • Analysis of zombies can help trace back to masters

◆ Compromised host often has a root backdoor
  • E.g., attacker runs TFN masters through root shell
  • Standard service on a non-standard port or standard port associated with a different service
  • Attacker connects from yet another machine

◆ Stepping stone: compromised intermediary host used by attacker to hide his identity
General Principle

- Find invariant or at least highly correlated characteristics of network links used by attacker
- Leverage particulars of how interactive traffic behaves
Indirect Stepping Stones

- Indirect stepping stone: “A-B ... C-D” vs. “A-B-C”
Timing Correlation of Idle Periods

- **Idle period** = no activity for $\geq 0.5$ sec
  - Consider only when idle periods end to reduce analysis possibilities
- Two idle periods are considered correlated if their ending times differ by $< 80$ms
  - Works even on encrypted traffic!
- **Detection criteria**
  - $\#$ of coincidences / $\#$ of idle periods
  - $\#$ of consecutive coincidences
  - $\#$ of consecutive coincidences / $\#$ of idle periods
Failures

- Large number of legitimate stepping stones
- Very small stepping stones evade detection
  - Limits attackers to a few keystrokes
- Message broadcast applications lead to correlations that are not stepping stones
  - Can filter these out
- Phase-drift in periodic traffic leads to false coincidences
  - Can filter these out, too