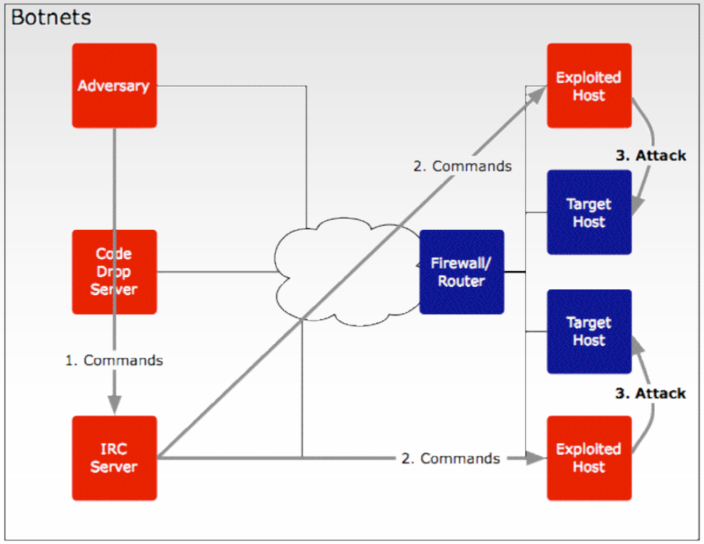
Identifying Rogue/Nefarious Applications

D. Lapsley, R. Walsh, T. Strayer
BBN Technologies
{dlapsley, rwalsh, strayer}@bbn.com

IMRG Workshop on Application Classification and Identification
October 3, 2007

The Problem

Botnets



Adversary

Code Drop Server

1. Commands

IRC Server

2. Commands

2. Commands

Firewall/ Router

Exploited Host

3. Attack

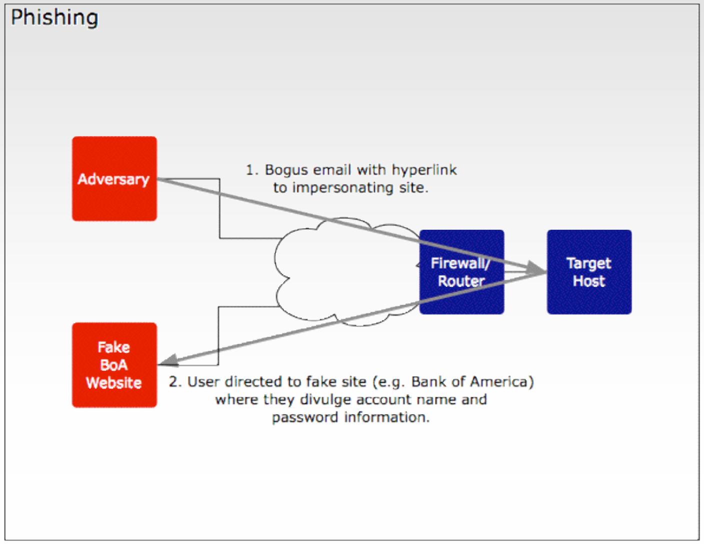
Target Host

Target Host

3. Attack

Exploited Host

Phishing



Adversary

1. Bogus email with hyperlink to impersonating site.

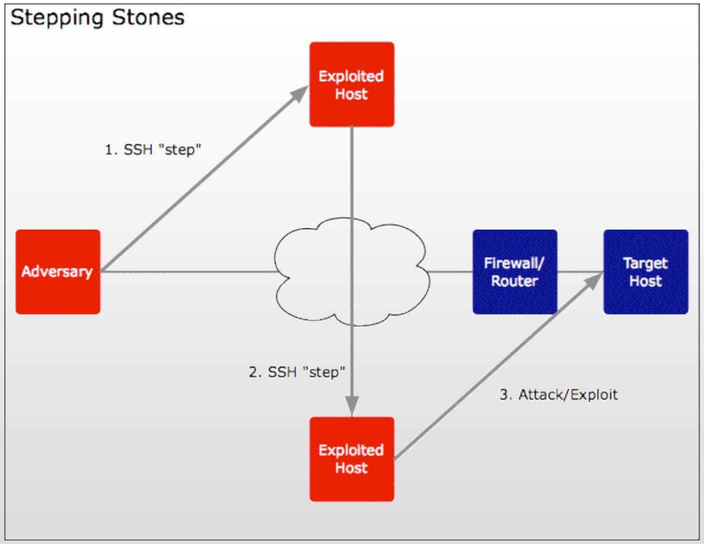
Firewall/ Router

Target Host

Fake BoA Website

2. User directed to fake site (e.g. Bank of America) where they divulge account name and password information.

Stepping Stones



Exploited Host

1. SSH "step"

Adversary

Firewall/ Router

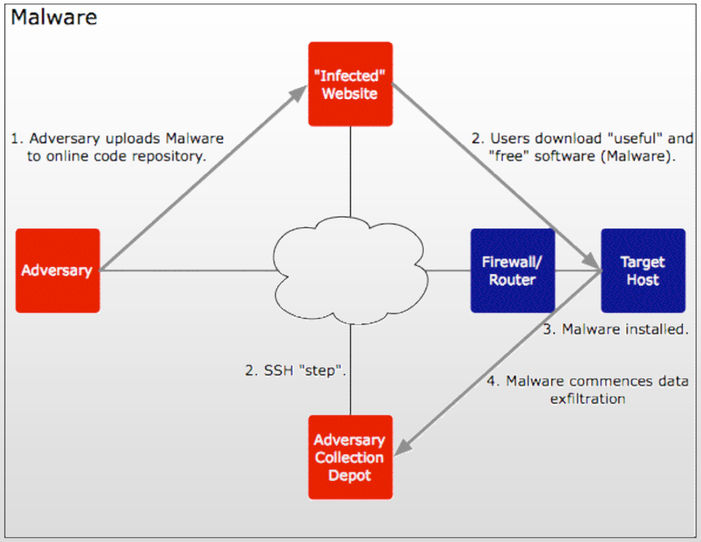
Target Host

2. SSH "step"

3. Attack/Exploit

Exploited Host

Malware



"Infected" Website

1. Adversary uploads Malware to online code repository.

2. Users download "useful" and "free" software (Malware).

Adversary

Firewall/ Router

Target Host

3. Malware installed.

2. SSH "step".

4. Malware commences data exfiltration

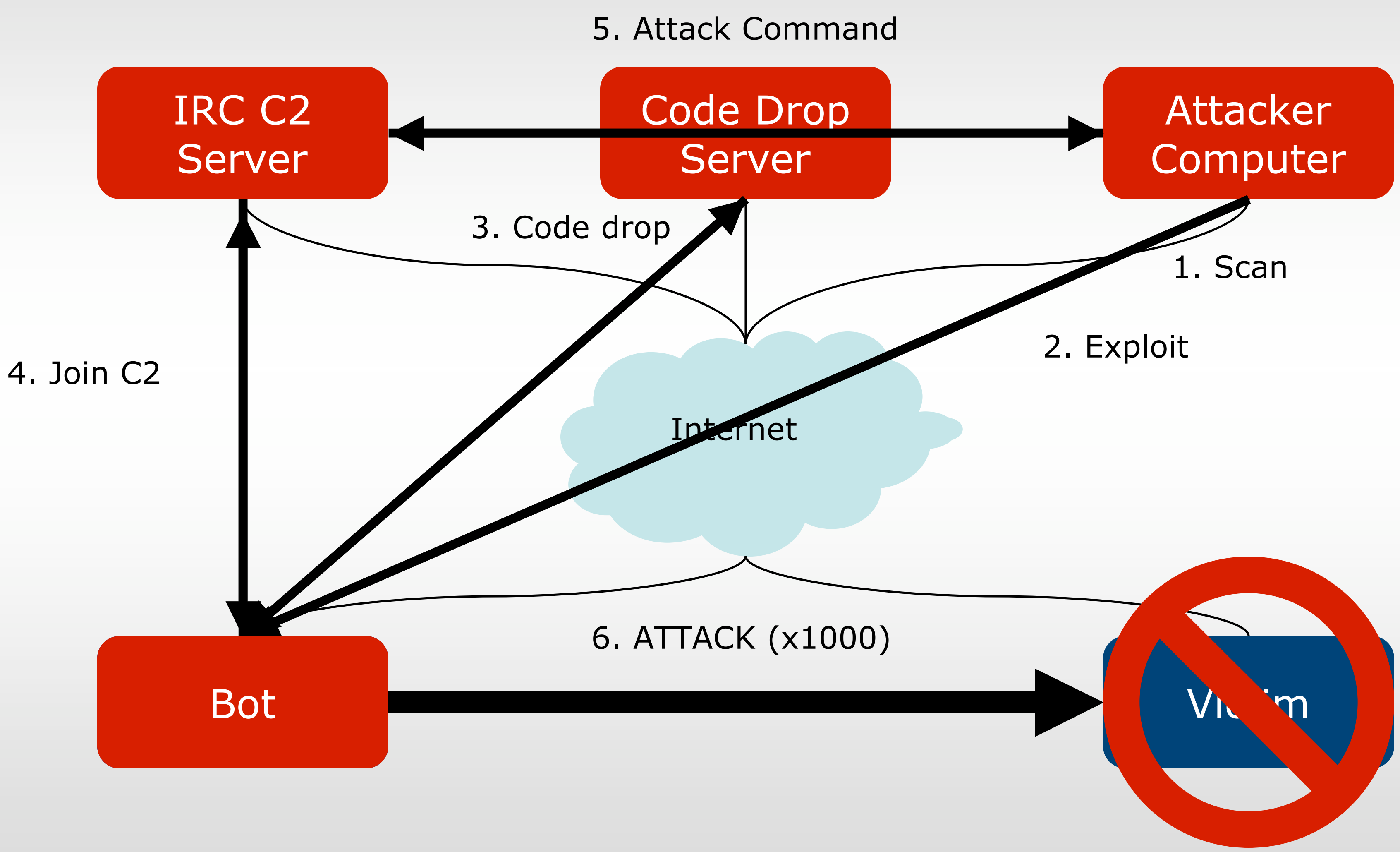
Adversary Collection Depot

Prevalence and Impact*

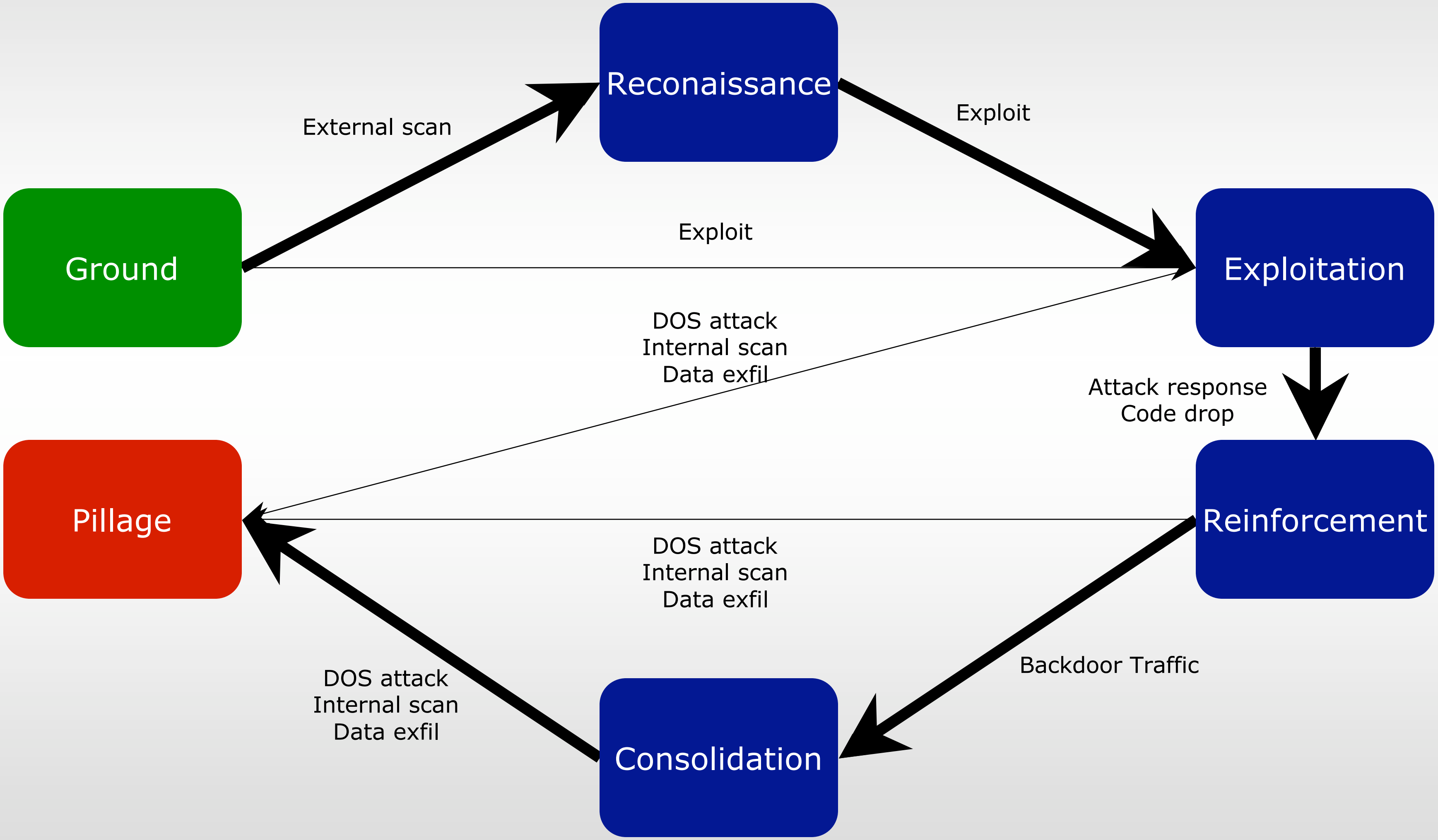
	Prevalence	Impact
Botnets	⇐ 5.029 million **distinct** botnet computers observed ▼ ⇐ 52,771 **active** computers/day ▼ ⇐ 4,622 bot **C2** servers ▼	⇐ Denial of Service, ⇐ Exfiltration of Sensitive Data, ⇐ 3rd Party Attacks
Phishing	⇐ 196,860 **unique** messages ▲ ⇐ 1,088 **unique** messages**/day** ▲ ⇐ 2.3 billion **blocked** attempts ▲	⇐ Exfiltration of Sensitive Data, ⇐ Destruction of data, etc., ⇐ Attack Vector for Botnets, Worms, Viruses, etc.
Stepping Stones	?	⇐ Obfuscation of Attack source
Malware	⇐ 212,101 **new** malicious code threats ▲ = 0.43% of all **spam** ▼ ≈ 0.26% of all **email** ▼	⇐ Exfiltration of Sensitive Data, ⇐ Destruction of data, etc., ⇐ Attack Vector for Botnets, Worms, Viruses, etc.

* Symantec, “Internet Security Threat Report XII: September 2007”

Botnet Operation



Botnet Attack Reference Model*



* R. Bejtlich, “The Tao of Network Security Monitoring”

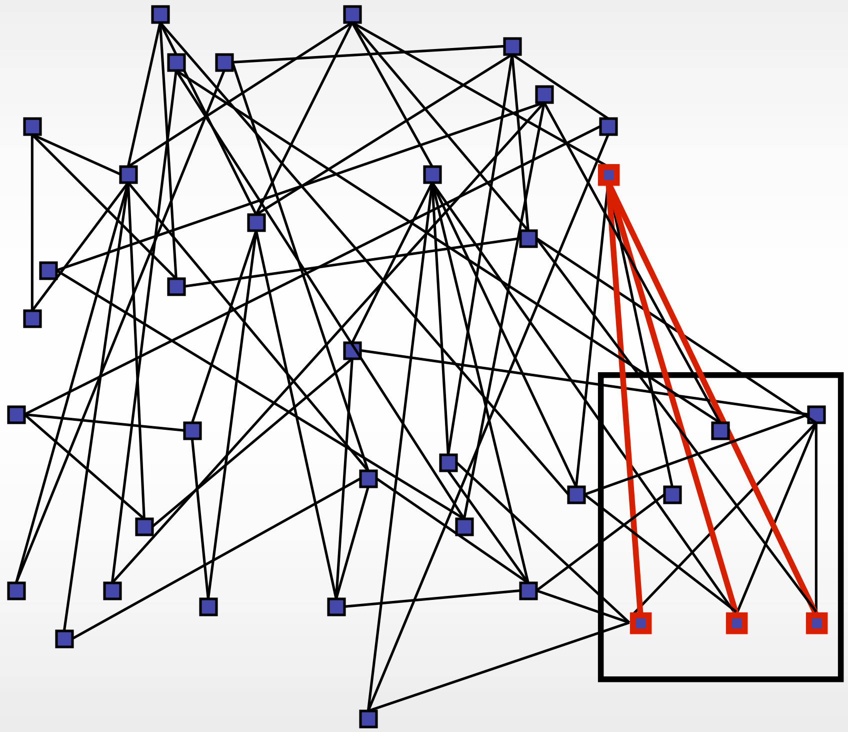
How to Catch a Botnet

- Variety of methods used to detect botnets
  - Use snort to examine payloads for IRC commands
  - Monitor free DDNS hosting services for instance
  - Construct Honeynets to surreptitiously join a botnet
  - Use host-based scanning software to examine hosts for rootkits, trojans, and other malware
  - Analyze traffic for patterns and correlations
- Each method has strengths and weaknesses
- Our work concentrates on traffic analysis

Current Approaches and Limitations

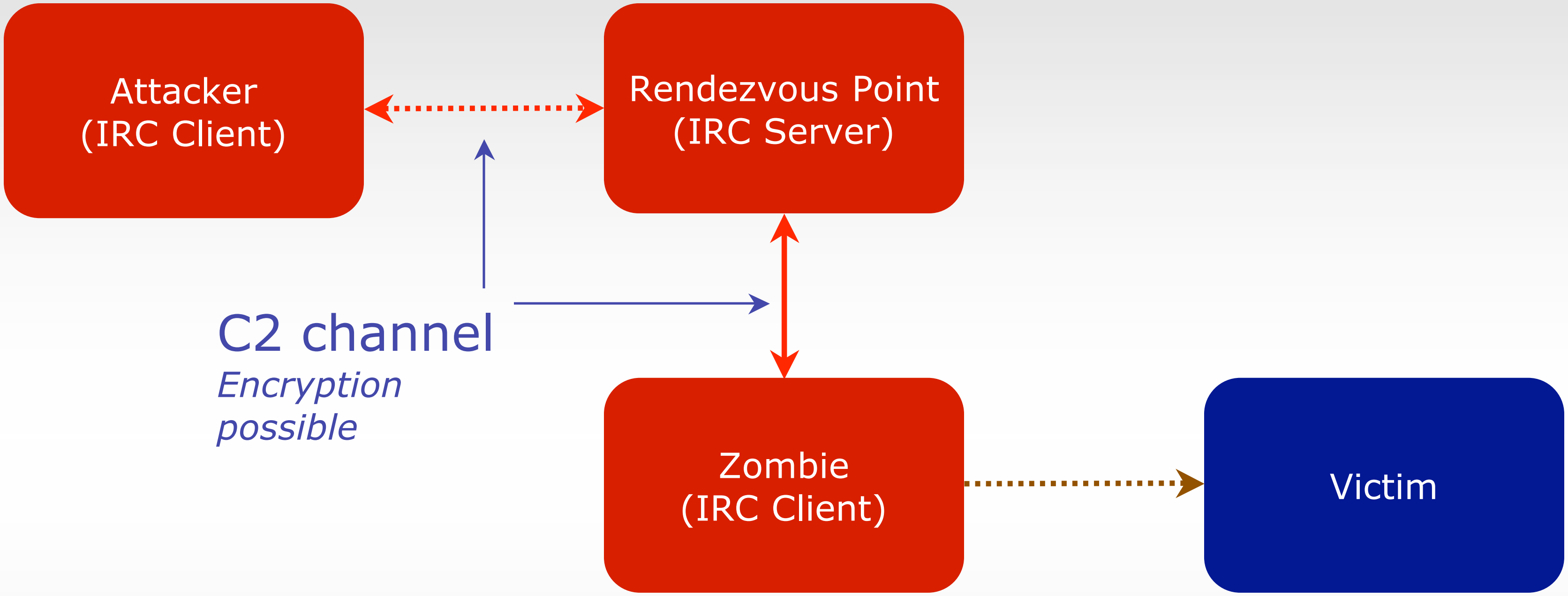
- Anomaly Detection:
  - Flags statistically anomalous data as a potential intruder or network attack
  - Limitations: training, obfuscation, false positives
- Signature/Pattern Recognition:
  - Intrusion Detection Systems (IDSs) inspect packet headers and contents and then match them against their database of signatures
  - Limitations: rate of new threats, window of exposure
- Reputation-based databases:
  - Databases identify “bad” and “good” websites/URLs
  - Application software does a database lookup for each URL request and allows or blocks request based on reputation of website
  - Limitations: window of exposure, no predictive capability, DOS effect
- Data fusion and event correlation:
  - Still early days
  - Promising “holistic” approach to network attack detection
  - Limitations: mechanisms for fusion and correlation still being developed and tested.

Traffic Analysis Botnet Detection



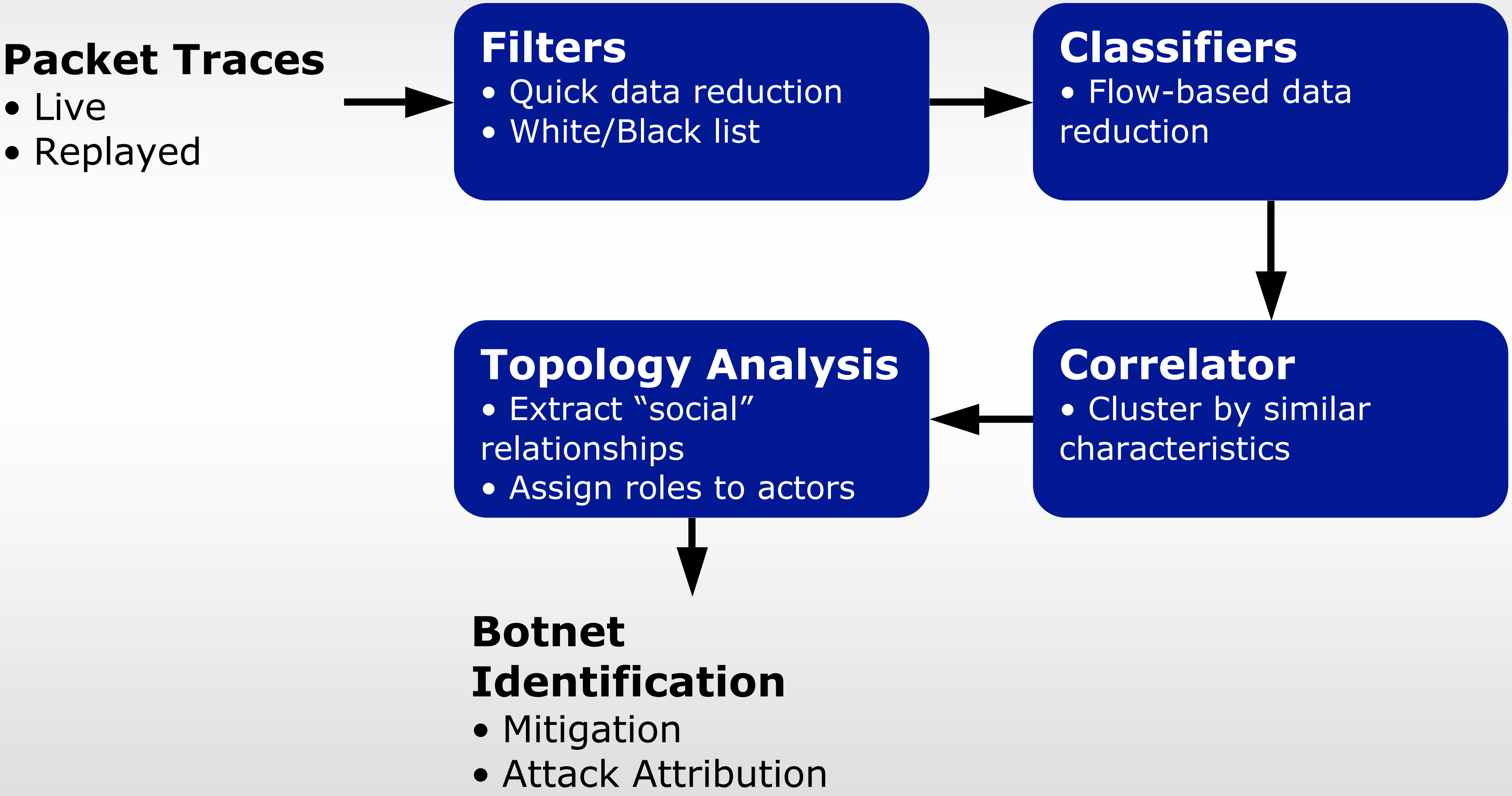
1. Monitor traffic within a region
2. Filter out and classify unlikely flows
3. Correlate flows to form a cluster
4. (Exchange with other monitors to widen the cluster)
5. Analyze the social aspects to piece together the botnet structure

Command and Control



- IRC is still dominant C2 technique
- We exploit certain IRC characteristics to exclude unlikely traffic and to discover botnet clusters
- As botnet C2 infrastructures change, we must continue to discover fundamental characteristics

Processing Pipeline Overview



Packet Traces
- Live
- Replayed

Filters
- Quick data reduction
- White/Black list

Classifiers
- Flow-based data reduction

Correlator
- Cluster by similar characteristics

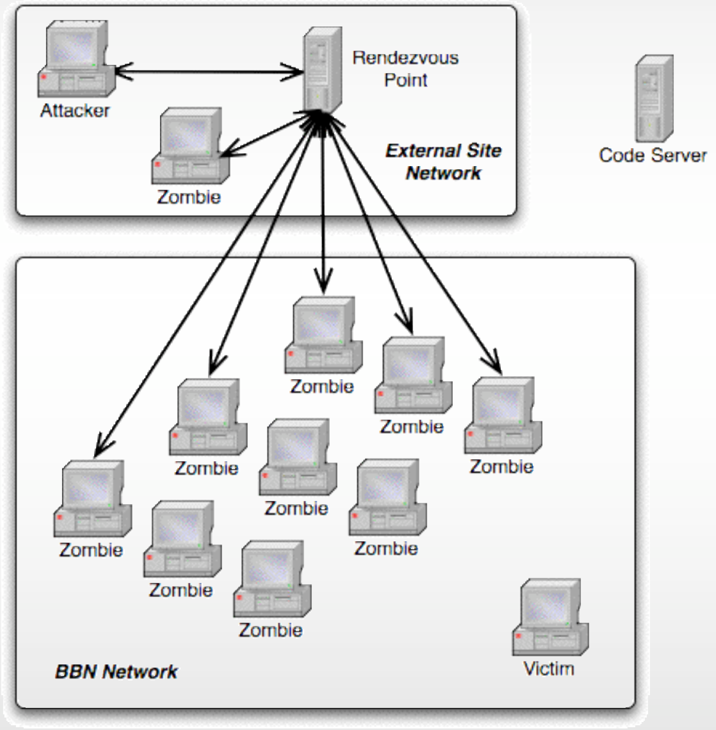
Topology Analysis
- Extract "social" relationships
- Assign roles to actors

Botnet Identification
- Mitigation
- Attack Attribution

Raw Packet Traces (Haystack)

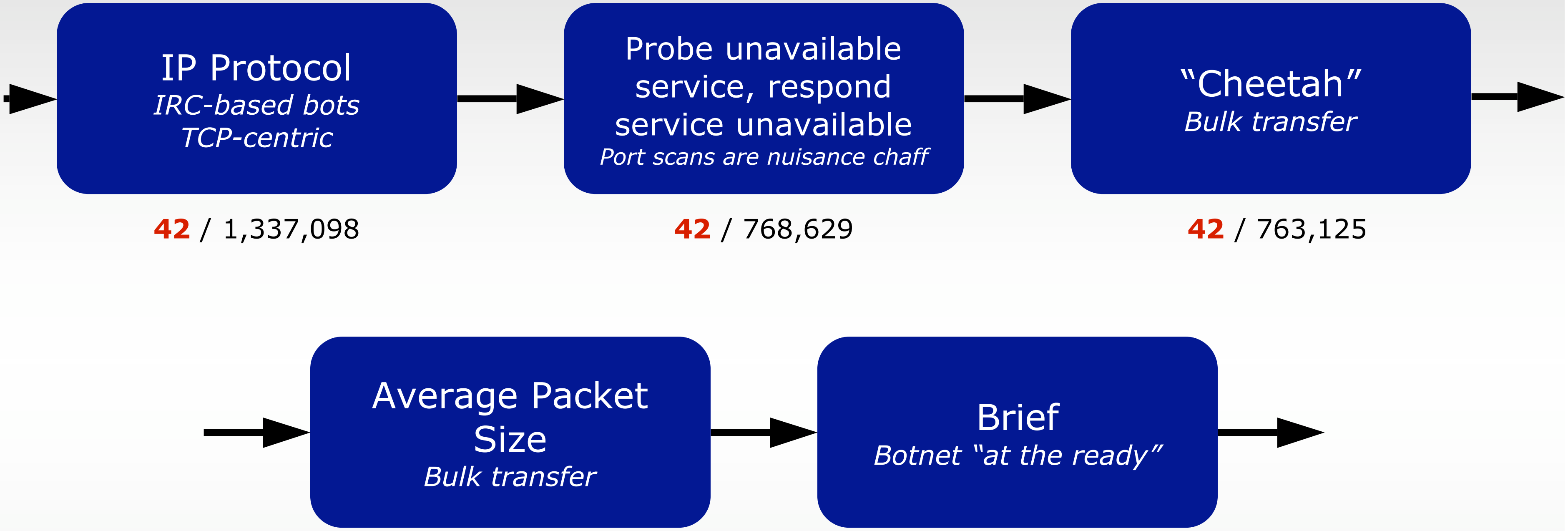
- TCP/IP packet header traces were acquired from Dartmouth Campus Wireless
  - A “CRAWDAD” data set
  - Variety of locations (dorm, library, academic buildings)
  - Gathered Nov 1, 2003 through Feb 28, 2004
    - About 9M total half-duplex flows in 4 months
    - 1.34M half-duplex flows in first 10 days
  - All IP addresses were obfuscated, no payloads

Botnet Traffic Traces (Needle)



- Built a botnet testbed
  - Need for "ground truth" traffic traces
  - Easily controlled
  - Reverse engineered and reimplemented "Kaiten" bot client; used standard BitchX server
  - 10 zombies, 1 controller, 1 server
- The botnet traces were overlaid with Dartmouth traces
  - 42 half-duplex botnet flows appropriately translated to the tenth day of Dartmouth data
  - 40 bot-server flows, 2 controller-server flows

Filters for Data Reduction



IP Protocol
IRC-based bots
TCP-centric

42 / 1,337,098

Probe unavailable service, respond service unavailable
Port scans are nuisance chaff

42 / 768,629

"Cheetah"
Bulk transfer

42 / 763,125

Average Packet Size
Bulk transfer

42 / 717,521

Brief
Botnet "at the ready"

42 / 36,228

- Quickly reduce data, making later (expensive) steps feasible
  - 37-fold reduction in data
- All **42** ground-truth botnet flows retained

Classification Technique

- Machine learning techniques have been shown to classify flows for QoS enforcement [Roughan’04, Moore’05]
- Approach
  - Label flows in training set as IRC/non-IRC based on port
  - Train classification model (Naïve Bayes, J48, Bayes Net)
  - Classify flows in testing set using WEKA machine learning tool
- Hope: Use “power” of conditional techniques (e.g., in Bayes Net) to classify flow

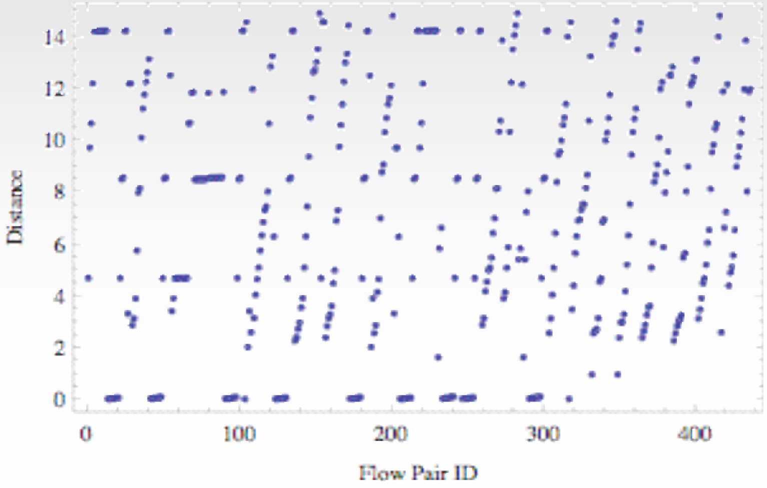
Classification Results

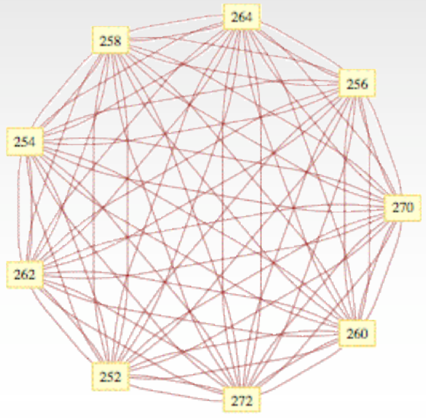
- Naïve Bayes performed best (planar slices, not conditional probabilities)
- Classification run on “filtered” traces
  - Reduced the remaining flows by nearly 70%
  - Surviving flows pruned down from ~36K to ~11K
  - 41/42 ground-truth botnet flows retained
- Accuracy very sensitive to
  - Classification scheme
  - Training set (didn’t train on botnet traces)
  - Attribute set

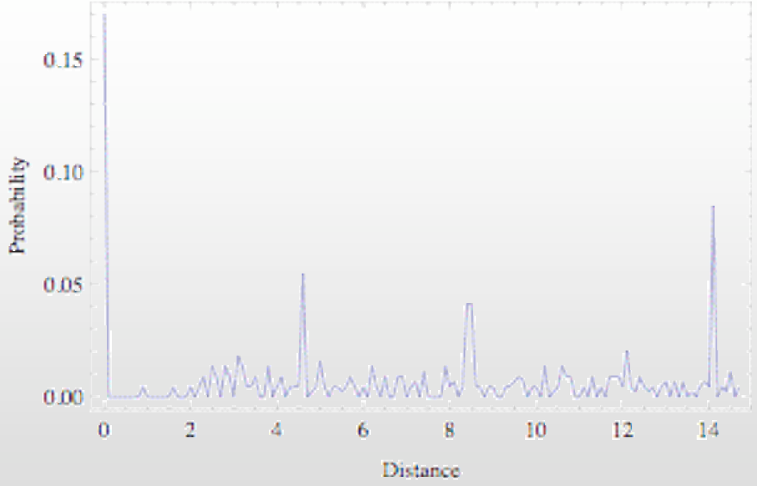
Flow Correlation

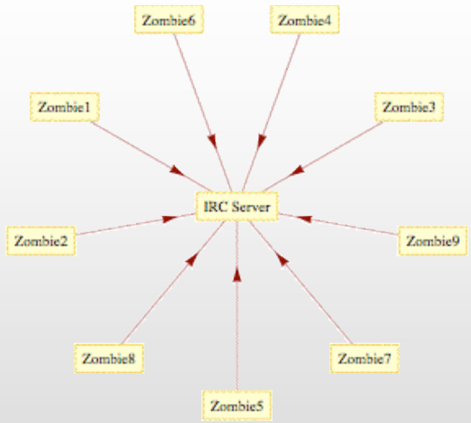
- Efficient centralized botnet control should form clusters of flows with similar behavior
  - E.g., receive packet from IRC server at about the same time, receive packets with similar interarrival times, …
- Picked a specific moment in time when botnet was active
  - 95 “filtered/classified” flows active at 15:30:00 on November 10, 2003
  - 22 were botnet flows active at that time
    - 20 of the 42 flows were finished before the test
    - 10 to bots, 10 from bots, 2 between controller and server
  - Rest were other flows that survived filters
- Did pairwise (NxN) correlation

Correlation Results









Future Directions

- Generalize detection for more sophisticated Bot architectures
- Generalize detection capability to other applications
- Combine traffic based analysis with other data sources
- Data fusion approaches