Correlations between quiescent ports in network flows

Joshua McNutt & Markus De Shon
{jmcnutt,mdeshon}@cert.org
September 21, 2005

© 2005 by Carnegie Mellon University

What is a quiescent port?

A TCP or UDP port not in regular use

- No assigned service
- Obsolete service
- Ephemeral port with no active service

© 2005 by Carnegie Mellon University

Port summary data

- Flows too detailed for some analysis
- Full flow data huge, slow interactive analysis
- Which flows are of interest?
- Therefore: Hourly summaries populate a database
  - # Flows
  - # Packets
  - # Bytes
  - Per port (TCP/UDP)
  - Per ICMP Type and Code
  - Per IP Protocol
  - “Incoming” and “Outgoing”

© 2005 by Carnegie Mellon University

Anomaly detection

There are many kinds of “anomaly detection”

Here we mean: statistical anomaly detection

Problem: Network data does not behave

- Self-similarity
- “Infinite” variance
- Not normal distributions

Problem: Data is noisy

- Vertical scanning
- Return traffic from web requests, outgoing email
- Other behavior masked

© 2005 by Carnegie Mellon University

Correlation

•Our realization:

•Vertical scanning leads to correlations between server ports

•Web & email return traffic leads to correlations between ephemeral ports

•Other kinds of activity may concentrate on only one port

- Horizontal scanning
- Backdoor activity
- Worms

© 2005 by Carnegie Mellon University

Robust correlation

Any anomaly detection method has a problem:

- What if the activity of interest occurs during the learning period?
- The model of “normal” is skewed

Solution: exclude the outliers

“Robust correlation”

- Exclude 5% most extreme outliers (Rousseew and Van Zomeren 1990)
- Calculate correlations based on remainder

© 2005 by Carnegie Mellon University

Robust correlation matrix

Take time series for ports (e.g. 0-1023)

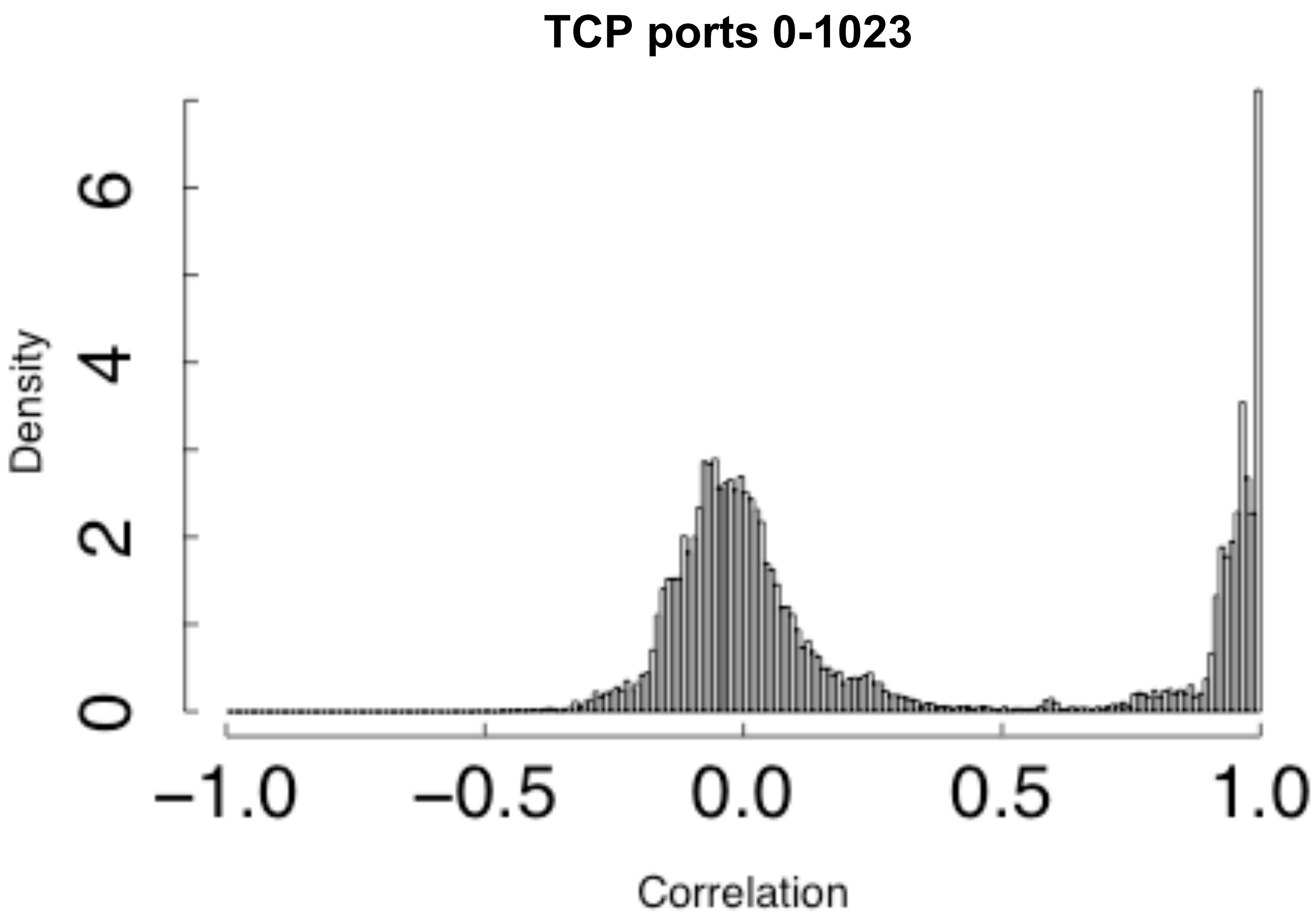
Calculate every robust correlation C(i,j)

C(i,j) is symmetric, and diagonal == 1

- C(i,i) == 1
- C(i,j) == C(j,i)

© 2005 by Carnegie Mellon University

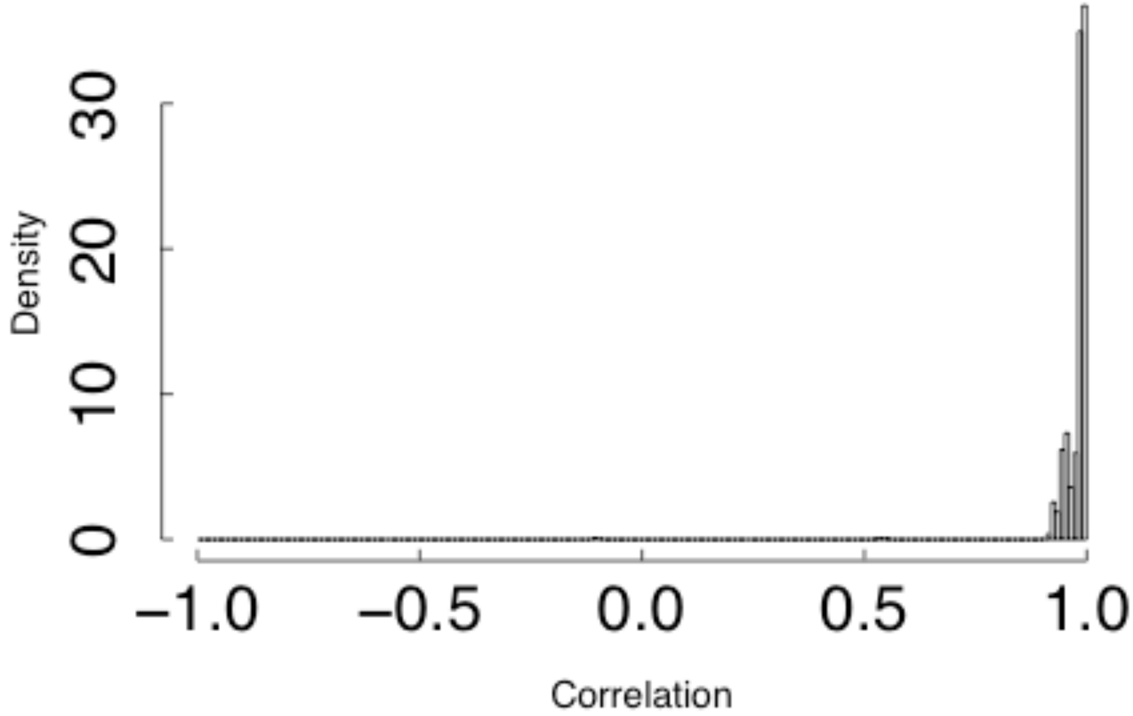
Robust correlation distribution



© 2005 by Carnegie Mellon University

Ephemeral port correlations (cont'd)

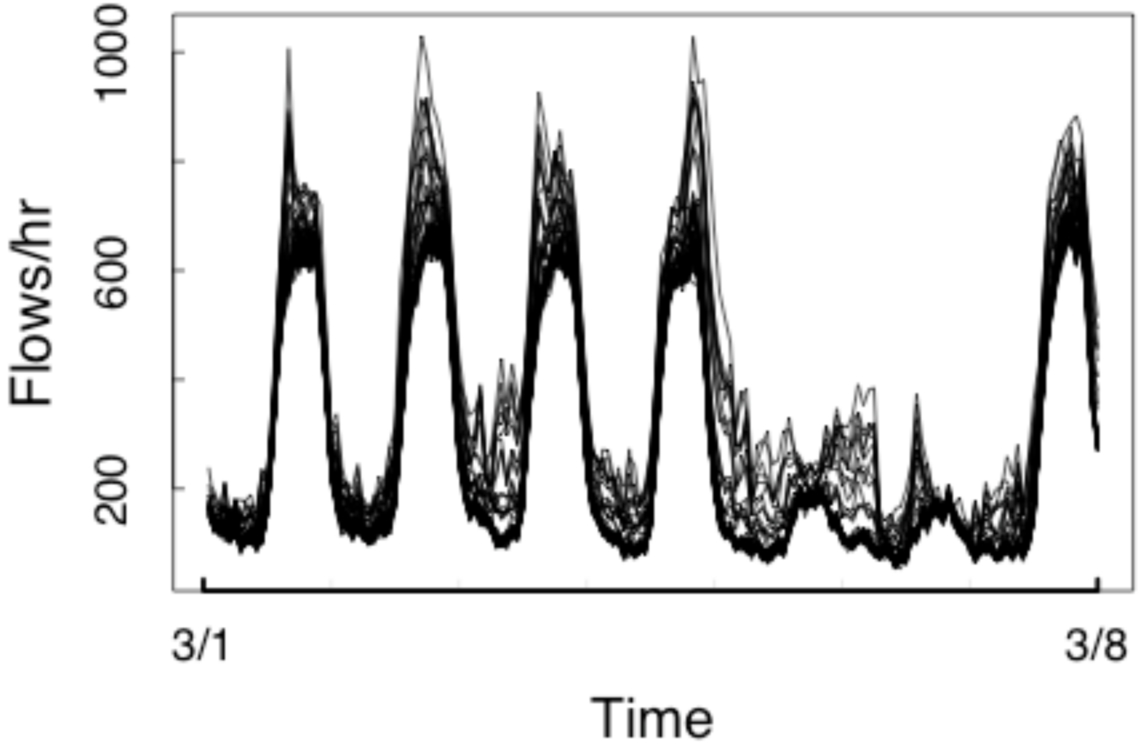
Robust correlation distribution (TCP/50000-51024)



© 2005 by Carnegie Mellon University

Ephemeral port correlations

50 high numbered ports



© 2005 by Carnegie Mellon University

Correlation clusters

Many correlated ports (indicated by ::)

If A::B and B::C, then A::C

Can we identify clusters A::B::C::D::…

Yes!

- For 0-1023, cluster of 133 ports
  - Could be higher with better data (need to include filtered traffic)
- For 1024+, nearly all ports are correlated
  - Large number of independent web browsers lead to well-behaved seasonality

Server ports

Ports 0-1023

Generally servers

Many unassigned/unused ports

Lots of filtering

Some obsolete services, possible source of threats

Ephemeral ports

Ports 1024-65535

A few servers

- Databases (Oracle 1521, MS SQL 1433/1434)
- Proxies (1080/8080)
- RPC services

Peer-to-peer

Backdoors (31337, etc)

Ephemeral ports for client services

- Request/response results in *two* flows

© 2005 by Carnegie Mellon University

The Method

Identify correlation cluster

Monitor all clustered ports, detect deviations

- Find median flow count for cluster, subtract from each port
- Significant number of flows above median → alert

Investigate deviations further

- Increased flows + increased hosts, intermittent → widespread horizontal scanning
- Increased flows + increased hosts, persistent → possible worm
- Increased flows, no increased hosts → localized activity, possibly still a threat

© 2005 by Carnegie Mellon University

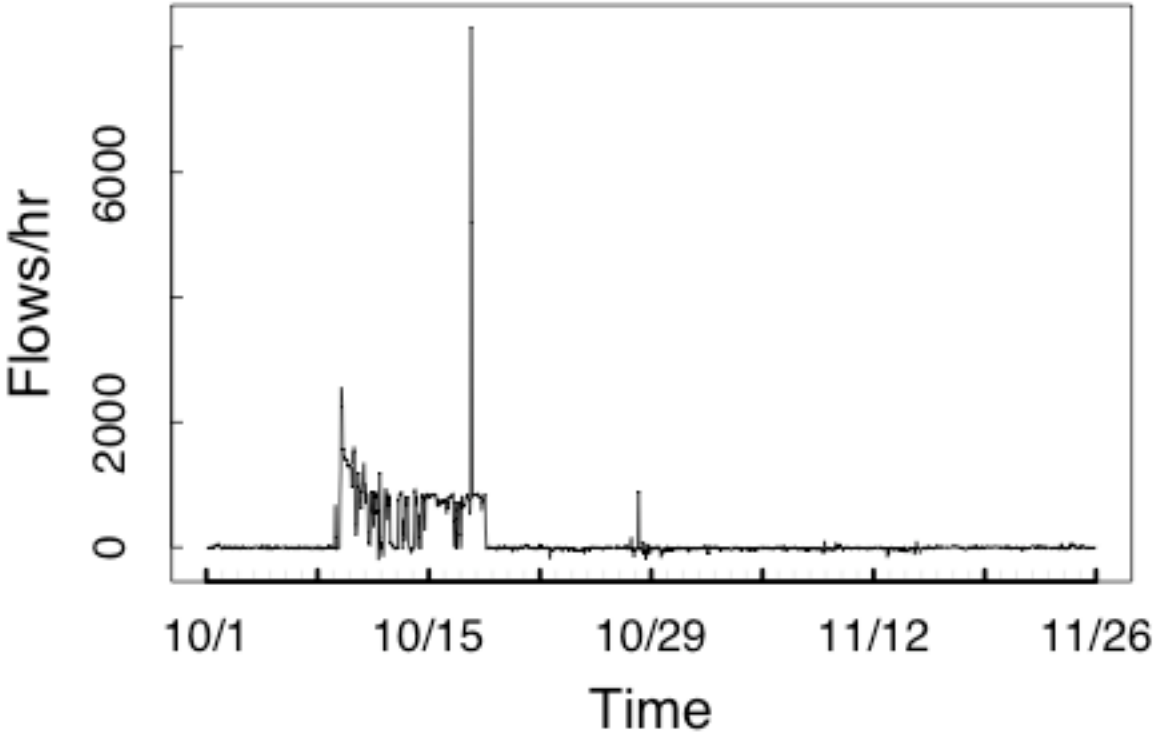
Case Study: 42/TCP

- Microsoft Windows Internet Name Service (WINS)
- Phasing out (replaced by Active Directory, DNS)
- Still present in Win2k3 Server
- Vulnerability announced Nov 25, 2004
- Scanning publicly announced Dec 12
- Could we have detected scanning earlier?

© 2005 by Carnegie Mellon University

42/TCP: Deviations from correlation

Before vulnerability announcement



© 2005 by Carnegie Mellon University

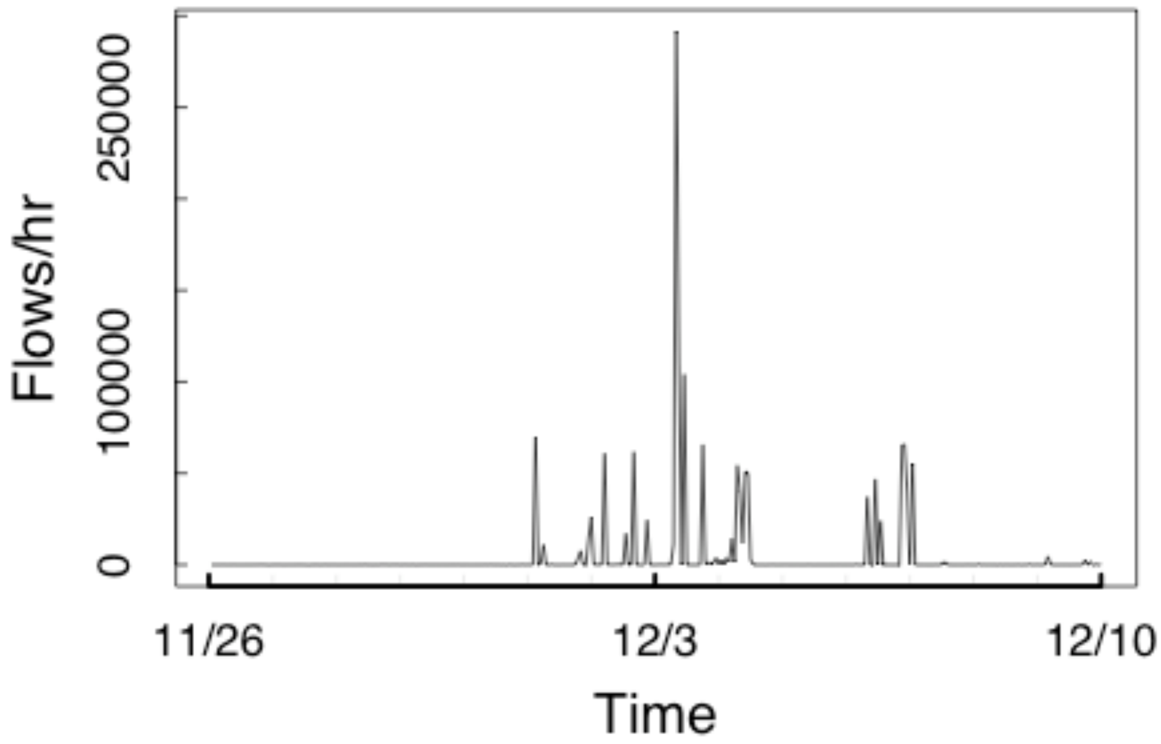
42/TCP: Deviations before vulnerability announcement

- Some deviations observed
- Always involved a small number of hosts (1 or 2)
- < 10,000 additional flows/hour
- No global activity indicated

© 2005 by Carnegie Mellon University

42/TCP: Deviations from correlation

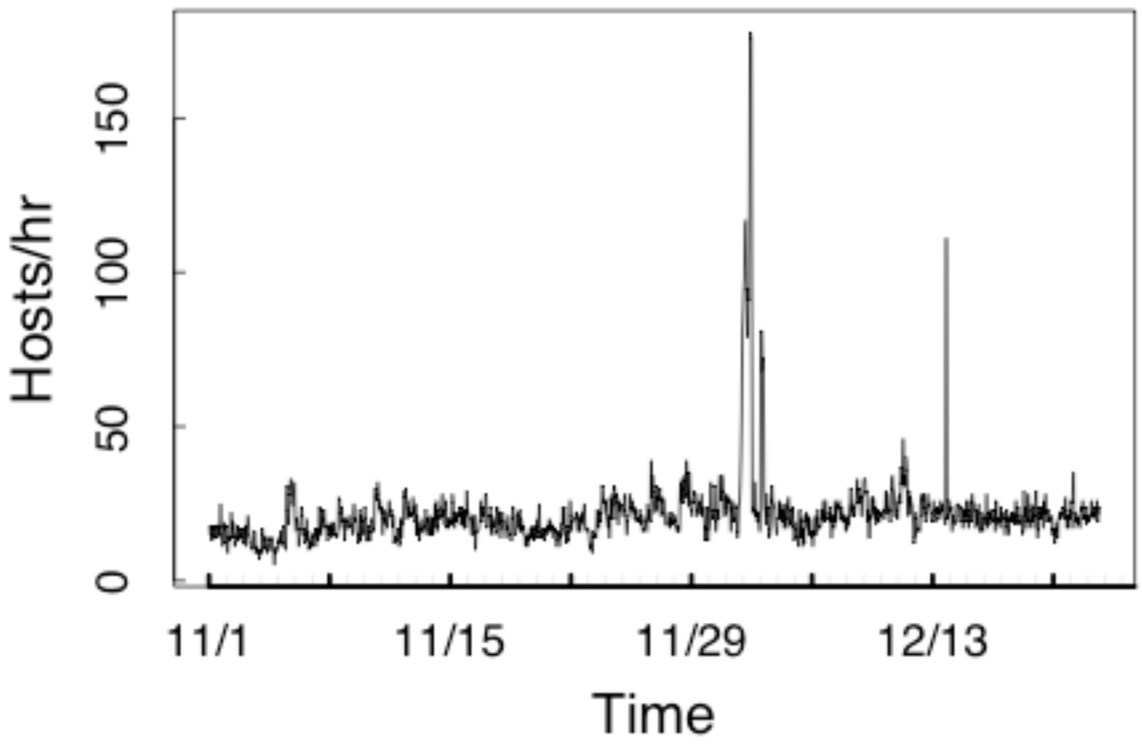
After vulnerability announcement, # flows/hr



© 2005 by Carnegie Mellon University

42/TCP: Deviations from correlation

After vulnerability announcement, # hosts/hr



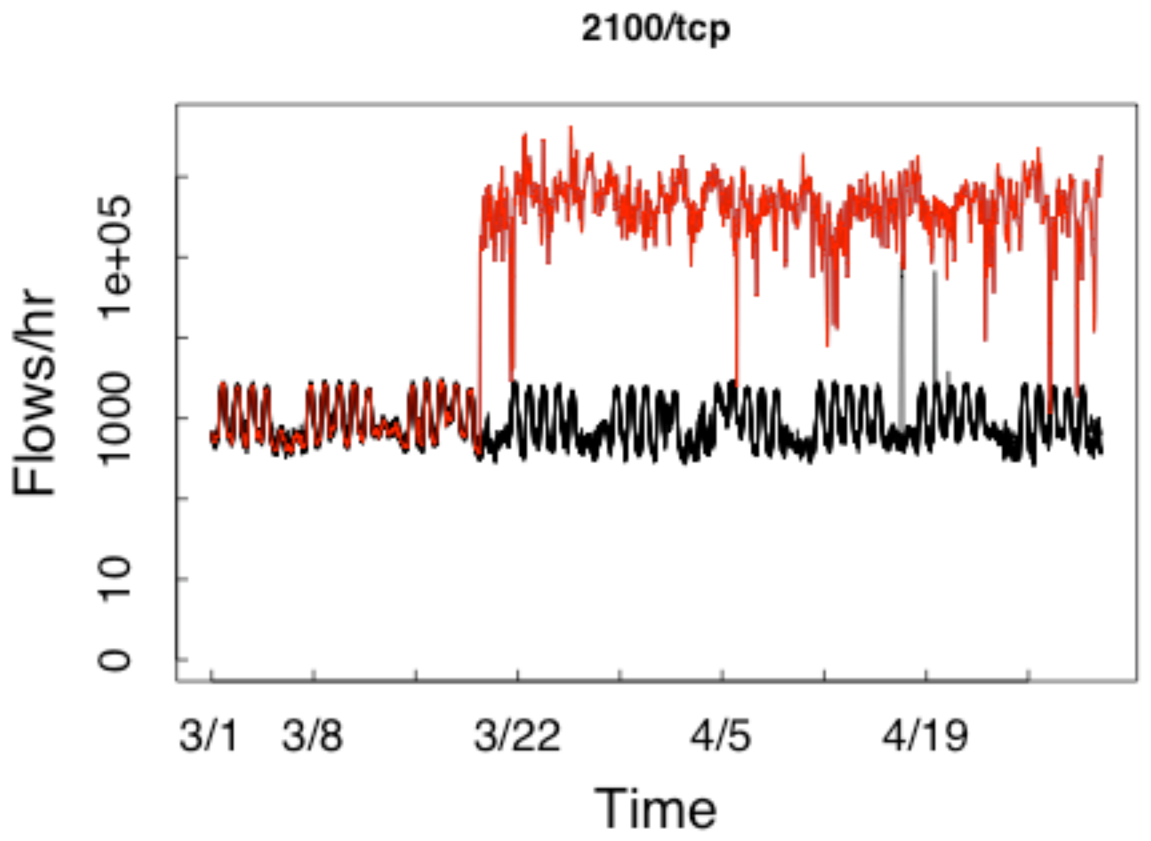
42/TCP: Deviations from correlation

After the announcement on 11/25

- Large increase in flows 12/1 2am (>100,000 additional flows/hr)
- Surge in #hosts/hr by 12/1 midnight
- Could have announced:
  - Scanning of port 42/TCP observed
  - Announce by morning of 12/2
  - Ahead of other announcement by 10 days

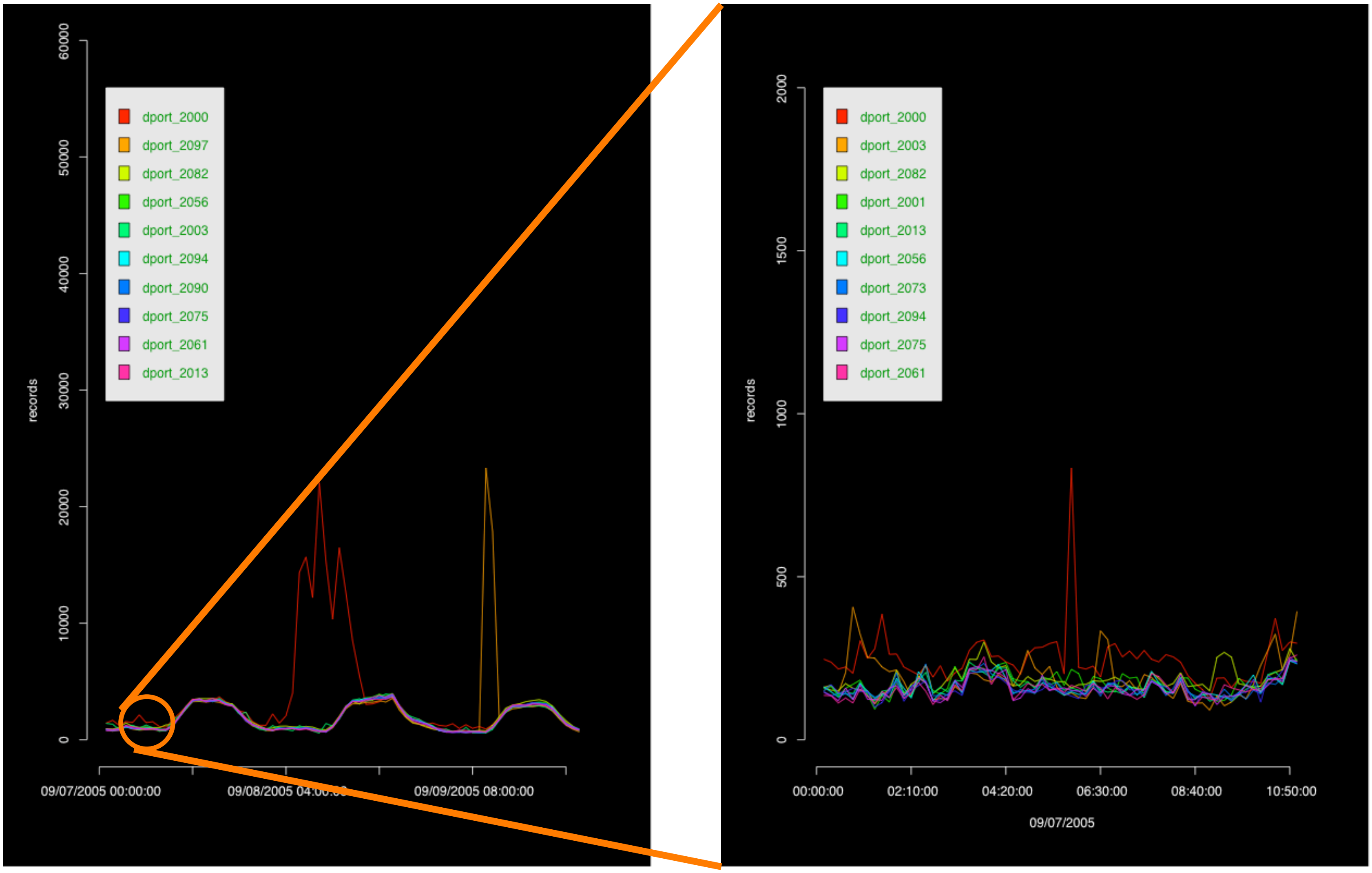
© 2005 by Carnegie Mellon University

Port 2100/TCP



© 2005 by Carnegie Mellon University

Interactive analysis



Future Directions

Median in sliding window of ports?

- Uncover attacks against ranges of ports

Unique number of sources, destinations

- ipsets?

Work on non-quiescent ports

- Some experiences with ephemeral ports (return traffic)
- Models will differ for different services
  - user-driven (e.g. web)
  - automated (e.g. ntp)

Flows vs. bytes vs. packets

- Peer-to-peer
- Information exfiltration

Automatic identification of backscatter (to be ignored?)

© 2005 by Carnegie Mellon University

Conclusions

Many ports highly correlated

- Vertical scanning (esp. server ports)
- Client activity responses (ephemeral ports)

Removing correlated activity exposes other activity

- DDoS backscatter
- Port-specific scanning
- Port-specific exploit attempts
- Worms

42/TCP real world example

- Clear signal
- Public announcement 10 days earlier

Automated method for focusing attention on specific ports

© 2005 by Carnegie Mellon University

CERT/NetSA

CERT/NetSA
Software Engineering Institute
Carnegie Mellon University
4500 Fifth Avenue
Pittsburgh PA 15213
USA

Web: http://www.cert.org/netsa

© 2005 by Carnegie Mellon University

Flow-based Analysis

A *flow* is a one-way network traffic instance

- Source ip and port → destination IP and port
- Corresponds to 1 side of a TCP session
- Aggregates UDP pseudo-sessions
- Times out

Example implementation: Cisco NetFlow

© 2005 by Carnegie Mellon University