

Columbia IDS Worminator Project

Sal Stolfo, Janak J Parekh
Michael Locasto, Angelos Keromytis
December, 2003

The need for Zero-day prevention

"Users Worry About 'Zero-Day' Attacks, Try to Secure Systems" Computerworld
(12/15/03); Vijayan, Jaikumar

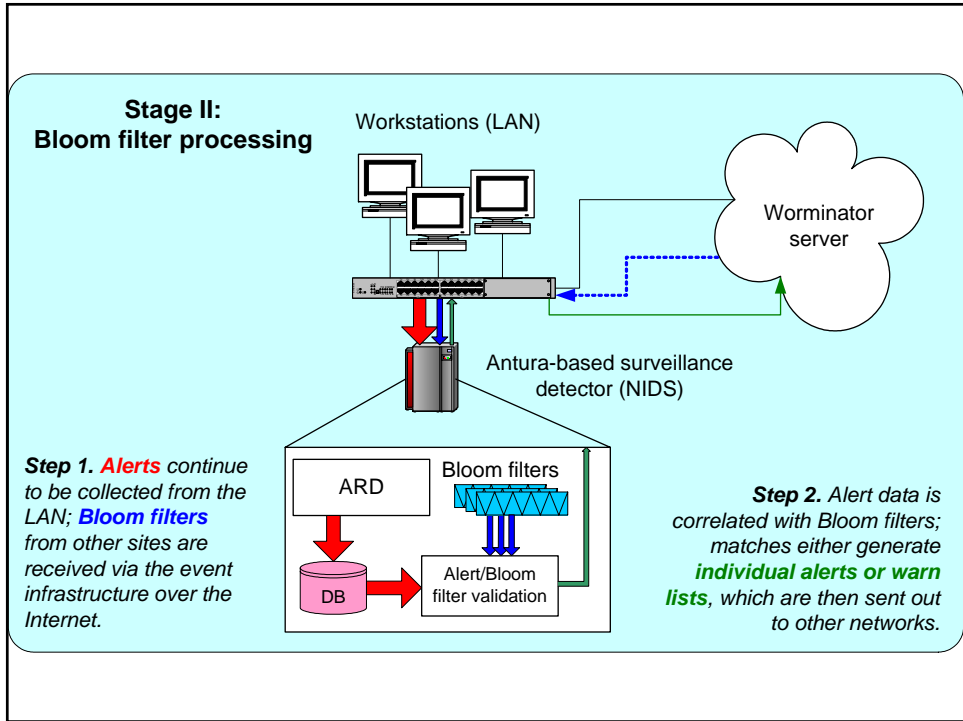
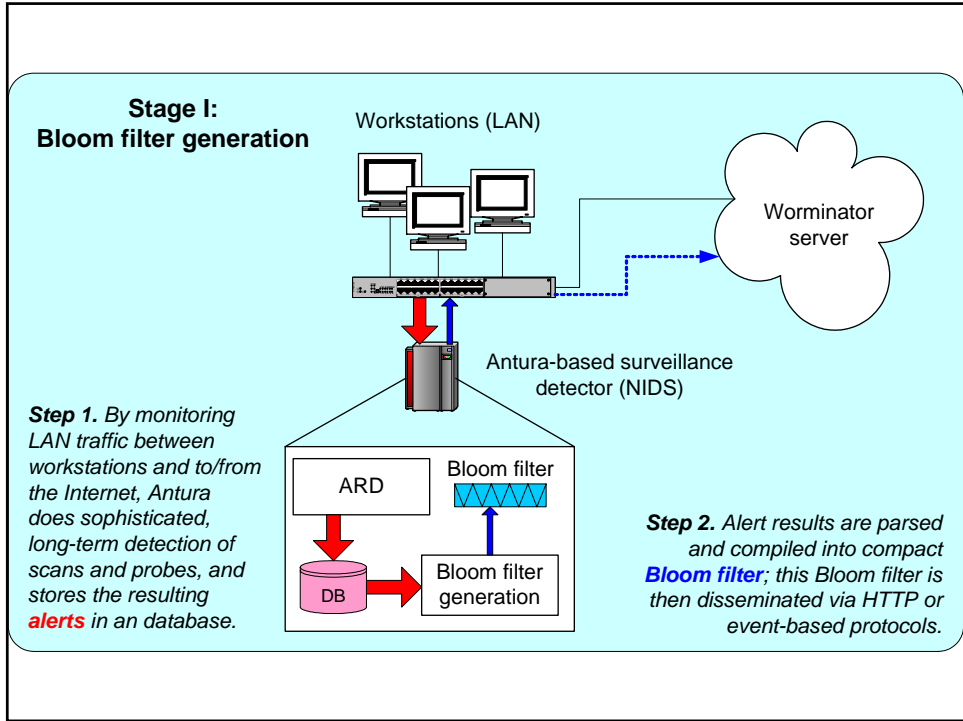
Companies are becoming increasingly concerned about the possibility that their systems will be attacked before software patches are made available. At the InfoSec 2003 Conference in New York last week, information technology managers indicated that so-called zero-day attacks have the potential to cause enormous damage to data security. Joseph Inhoff, LAN administrator for lighting equipment maker Lutron Electronics, said company management is concerned about a zero-day attack, adding that in attending the conference he wanted to learn whether automated patching software would be an effective way to secure systems. Industry experts say the amount of time it takes a hacker to exploit a software flaw has been reduced substantially. For example, the SQL Slammer worm appeared last January, eight months after the vulnerability in the SQL Server database was discovered; but last summer's Blaster worm came nearly one month after Microsoft issued a Windows patch. Industry observers believe hackers will eventually attack systems before flaws are disclosed and vendors release patches. A zero-day attack might force an unprepared company to shut down their systems and restart them.

Goal

- Individual IDses are “noisy”
 - Difficult to differentiate legitimate versus illegitimate traffic based on a narrow world view
- With the advent of worms and distributed attacks, possibility of *sharing* alert information
 - IDS watchlist correlation reduces noise, speeds detection of such propagating worms
- Sharing of alerts provides rapid detection to sites in a “pre-attack” stage
 - Share alerts and detected anomalous payloads
- LAN-based filters provide the means to eliminate a wide spread propagation thru an enterprise
 - An architecture that combines perimeter and interior defenses for worm prevention

Architecture for sharing alerts

- **Use *Bloom filters***
 - Compact, efficient, one-way data structures
 - Supports “insert” and “verify” operations
 - Allows correlation without releasing sensitive alert information
- **Generate and compare Bloom filters**
 - Use HTTP as a transfer protocol to a central server
 - Download nightly-generated Bloom filters and compare results across sites



Project status

- Code developed, compatible with Antura ARD v1.0
 - Written in Java, compatible with Antura's PostgreSQL database schema
 - Future versions will support other IDSes, including Snort (<http://www.snort.org>)
- In process of testing and rolling out to beta sites
 - Currently Columbia and Georgia Tech
- More information at <http://worminator.cs.columbia.edu>

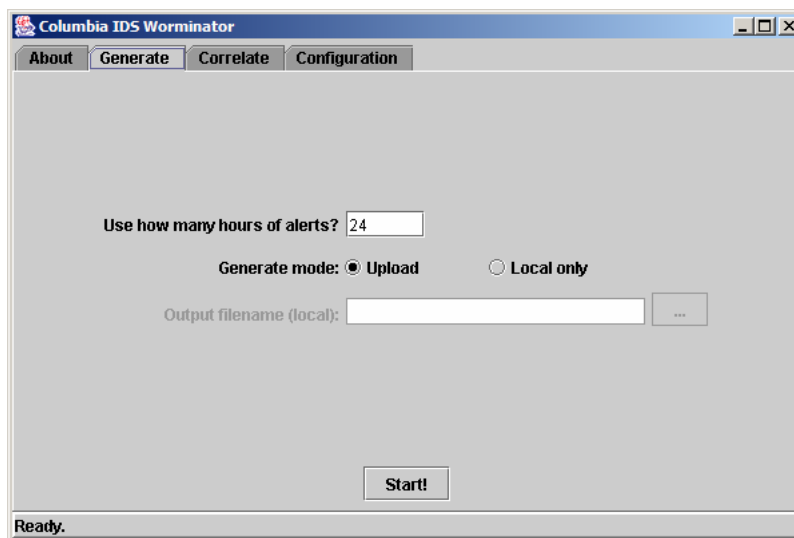


Figure 1. Worminator generation configuration. Bloom filter generation can be done on-demand or as a cron job.

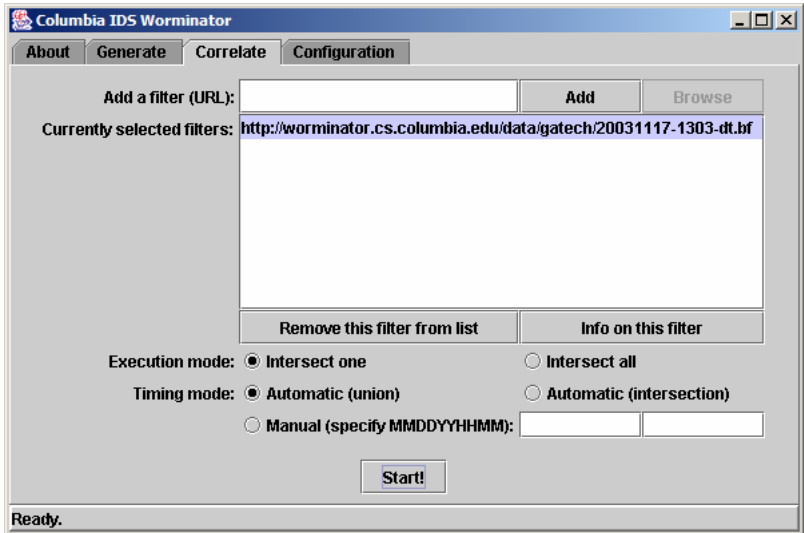


Figure 2. Worminator correlation configuration. A number of different options are available if multiple Bloom filters are used in tandem with a local alert database.

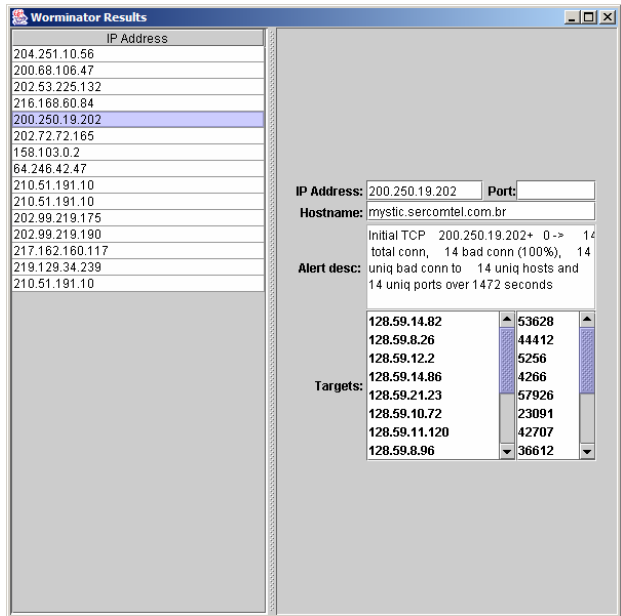


Figure 3. Example looking at just common source IPs. This example is based on data received at Columbia University and Georgia Tech. Individual detail is shown for one.

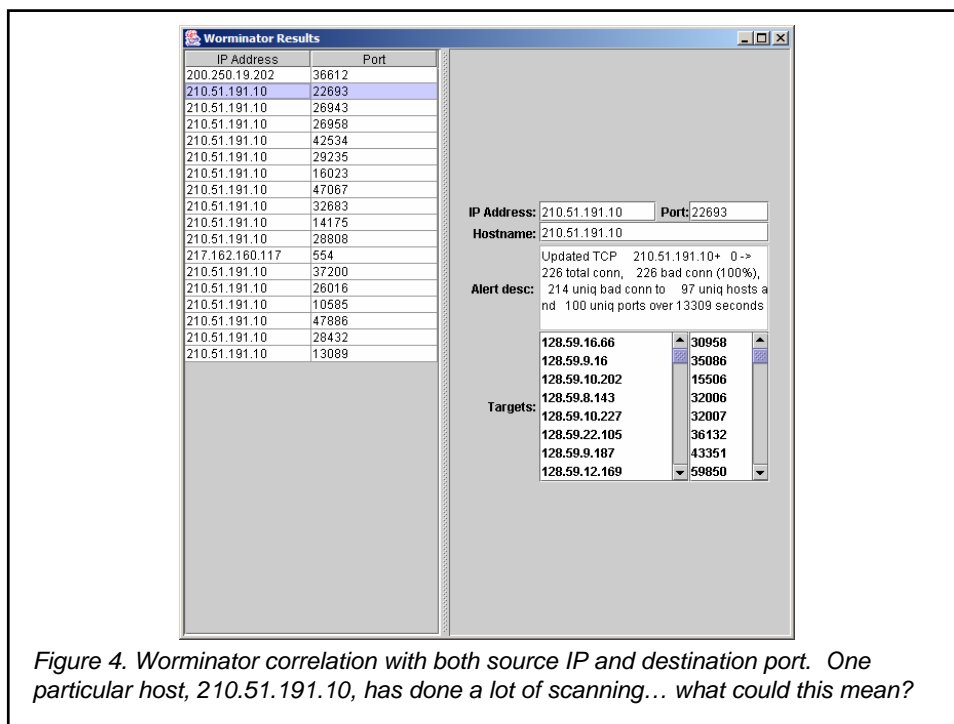


Figure 4. Worminator correlation with both source IP and destination port. One particular host, 210.51.191.10, has done a lot of scanning... what could this mean?

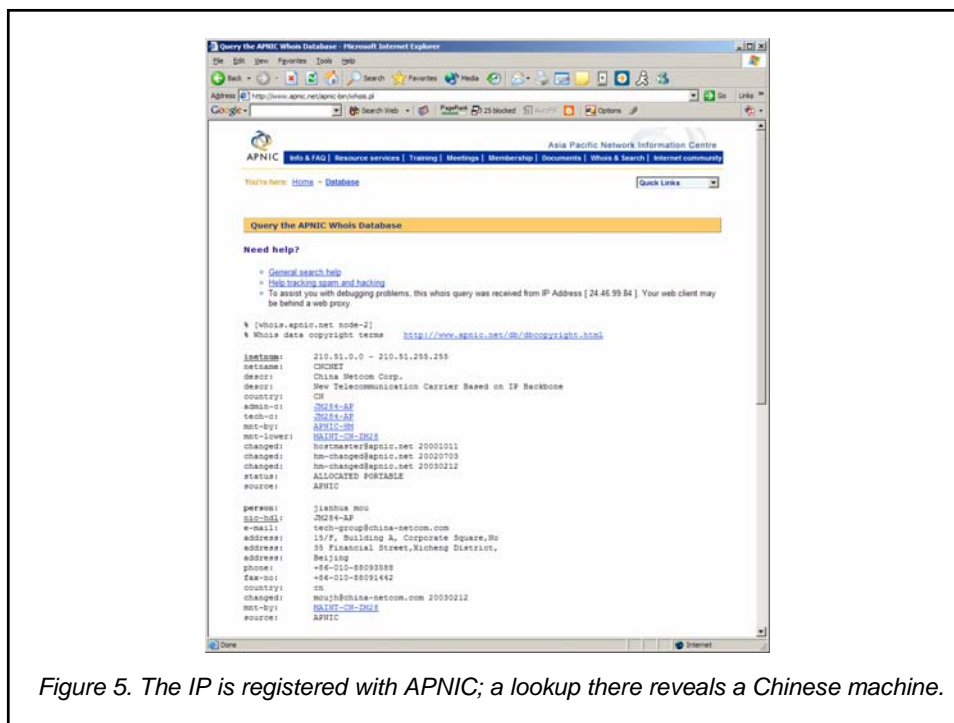
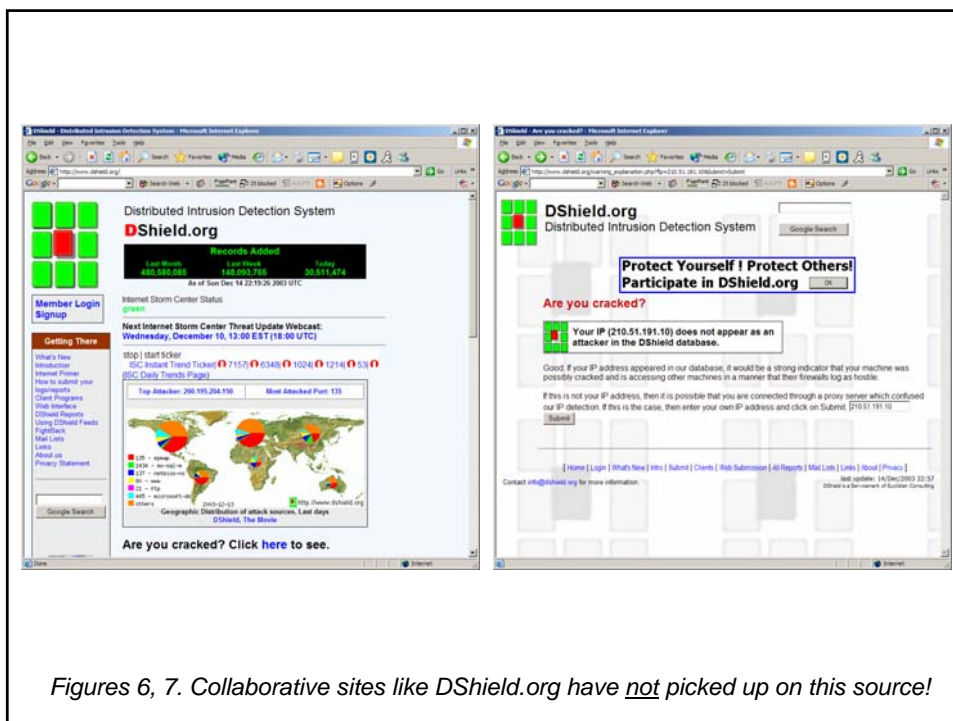
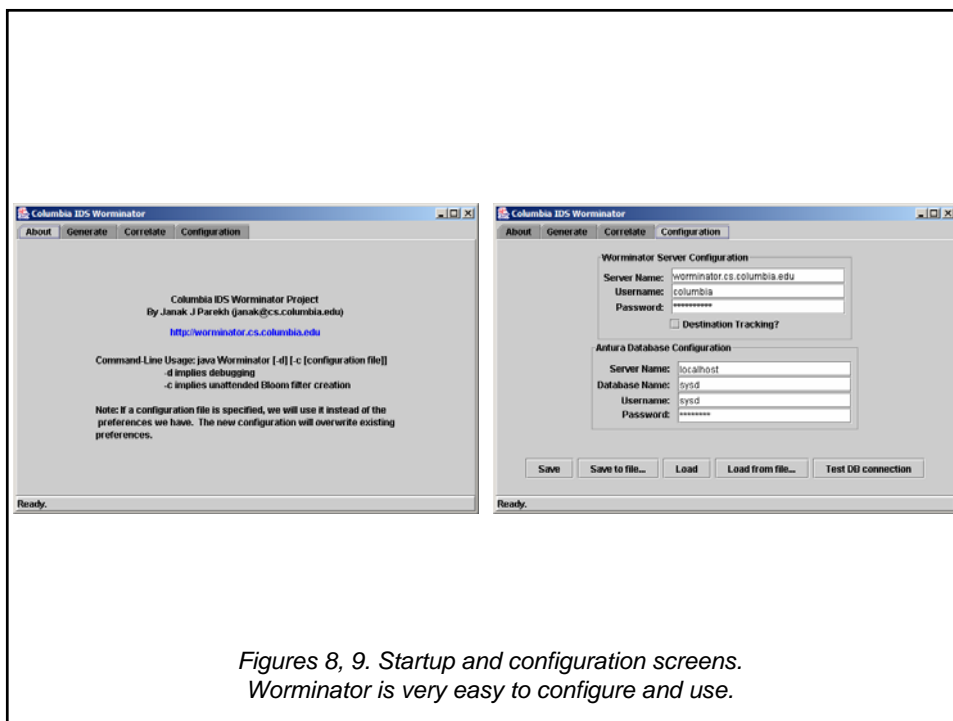


Figure 5. The IP is registered with APNIC; a lookup there reveals a Chinese machine.



Figures 6, 7. Collaborative sites like DShield.org have not picked up on this source!



Figures 8, 9. Startup and configuration screens. Worminator is very easy to configure and use.

Payload Anomaly Detection at the perimeter:
The PAYL sensor

Payload Analysis...Uncharted Territory

- Abnormal payload very difficult in High bandwidth environments
- Possibly quite feasible in distributed applications with lower bandwidth communication between communicating components
- Examples of network traffic payload analysis, 1-gram distributions...(Violations of Volume Distribution of Byte Values)

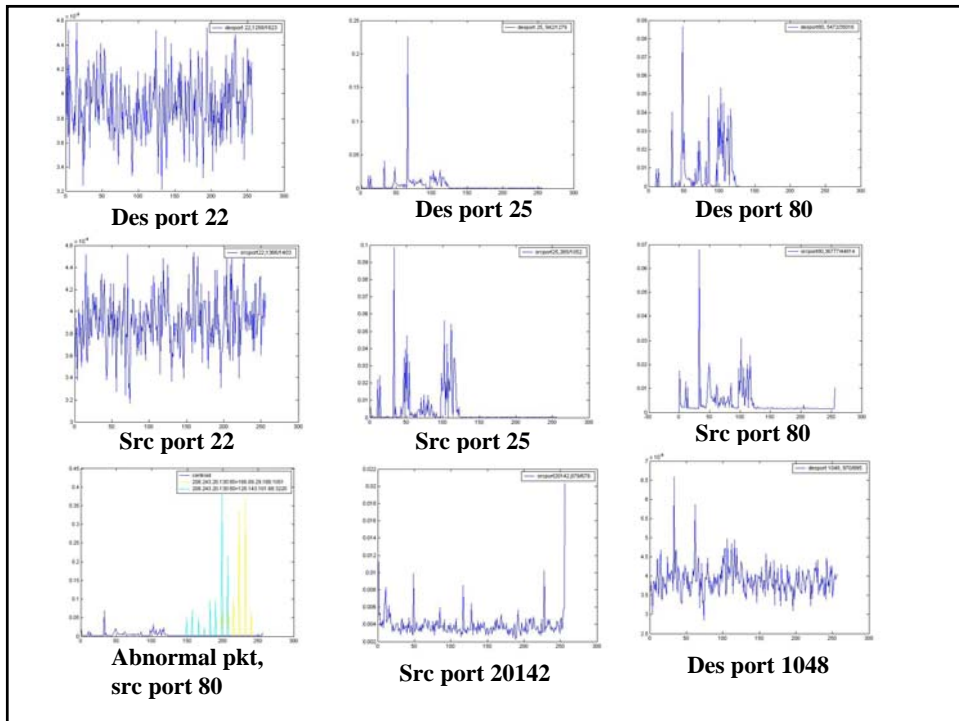
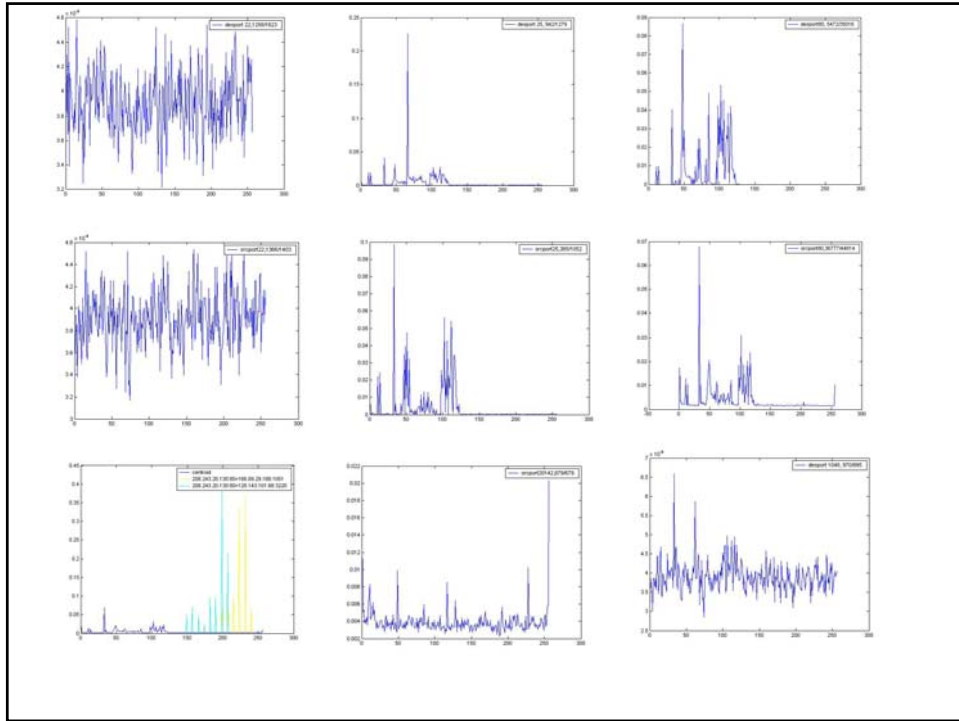
Prior Work on Payload Analysis... "Thumbprints"

- Haberlein [1992, 2002]
- Representation of "known" bad payloads (eg., Worm payloads)
- Hash coding of strings, "robust checksums" for "similar" strings and recently PCA vector model to represent similar strings

- What's different here:
 - PAYL Models NORMAL payload to detect ABNORMAL payloads
 - Attacks must be abnormal!
 - There is a plentiful supply of normal payloads to model, but "bad" payloads are available only when one is attacked!
 - Theoretically, this provides broadest coverage against any attack, not just those that have been seen already!

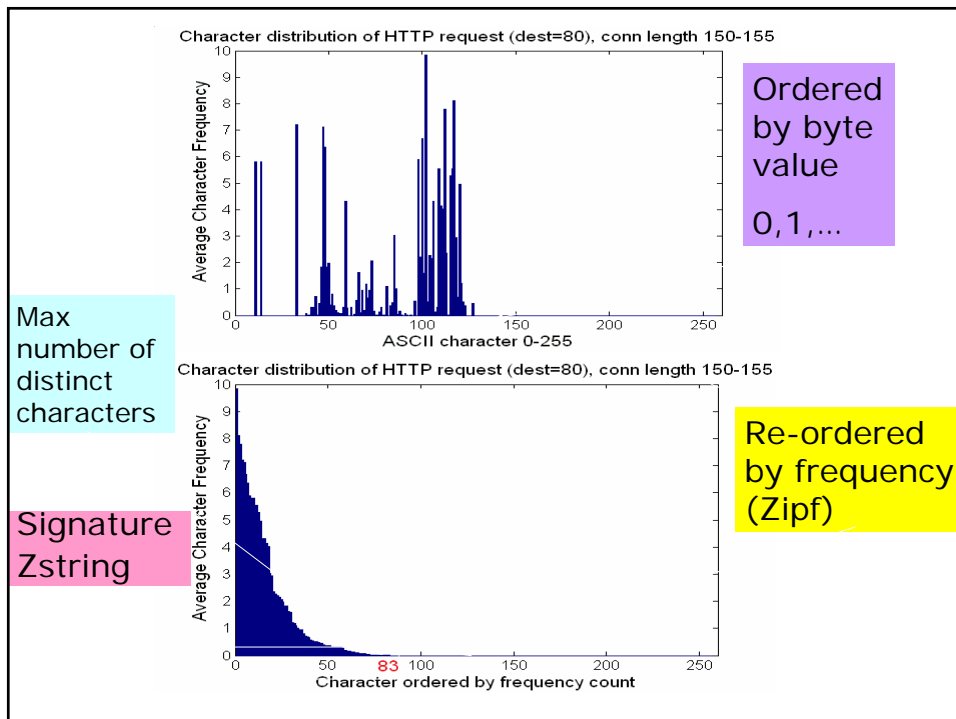
What's new?...

- Model Normal Payload rather than modeling Bad payload
 - There is far more normal data continuously available
 - Zero day bad payloads are "unknown" and unavailable for analysis
- Detect Abnormal Payload as uncharacteristic byte streams different than what is normal



Experiment on HTTP traffic

- Normal data
 - 24 hours HTTP traffic of INBOUND www.cs.columbia.edu,
 - 83272 request, 83298 response connections
 - Character distribution over different connection lengths
 - (Rank order Distribution converted to signature string)
- Test data: Code Red
 - (a) SecurityFocus report attack signature
 - (b) tcpdump <http://www.bofh.sh/CodeRed/index.html>



Signature string for HTTP request, length
150-155

```
eto $\alpha$  $\beta$ c/a lsrw:imnTupgbhH|0AdxEPU  
CG3*vF@_fyR,~24RzMk9=());SDWIj  
L6B7Z8%?Vq[]ONK+JX&`
```

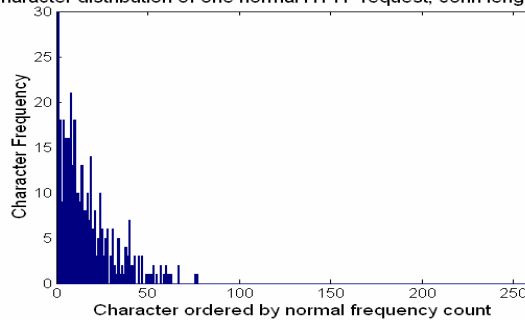
α : LF - Line feed

β : CR – Carriage return

Similar to a “footprint” (not thumbprint) for http
traffic of this length

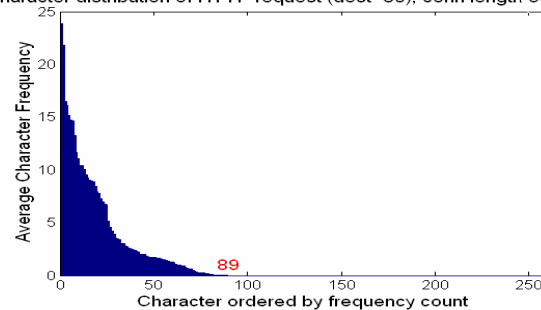
83 Distinct Characters in this distribution

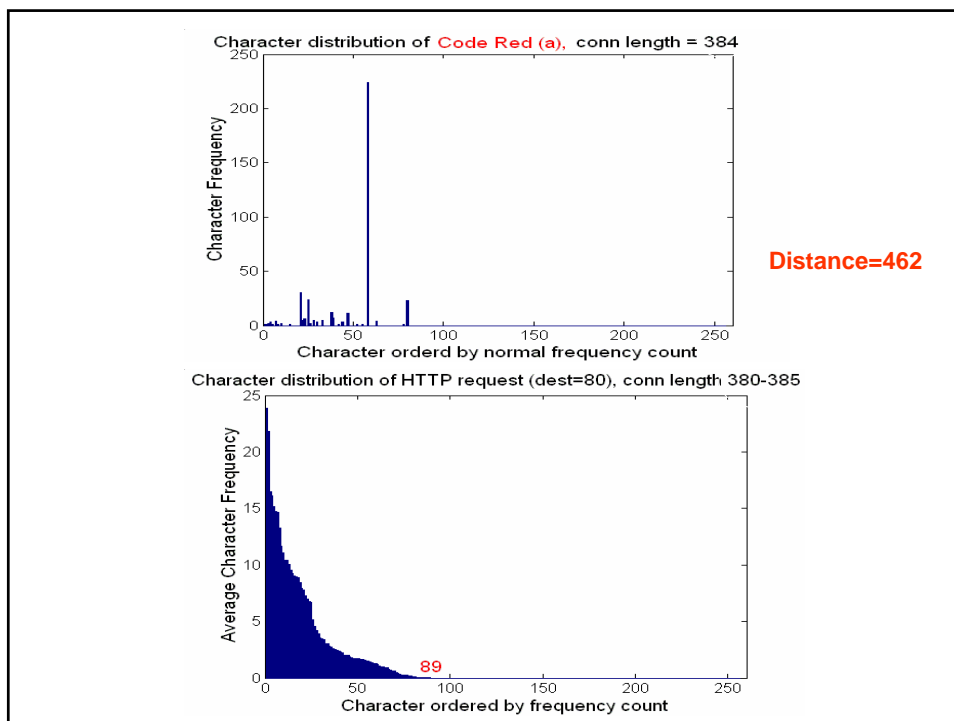
Character distribution of one normal HTTP request, conn length=382



Comparing
Distributions:
Mahalanobis
Distance=52

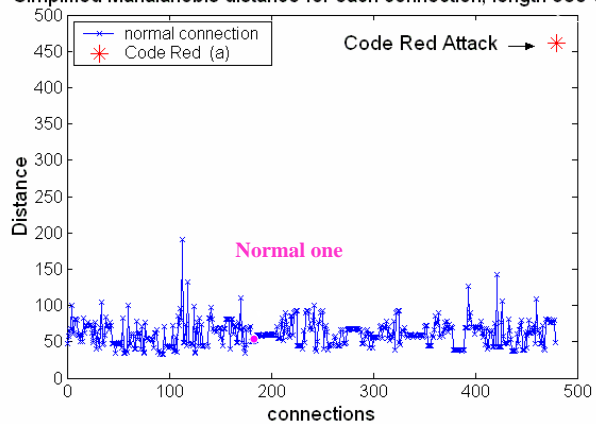
Character distribution of HTTP request (dest=80), conn length 380-385

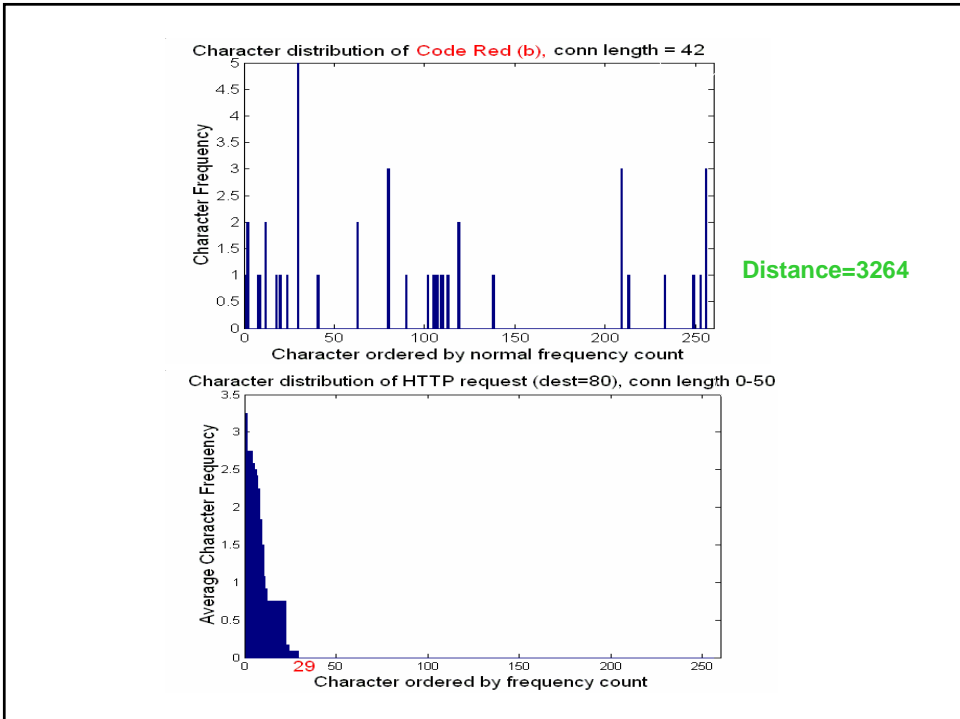
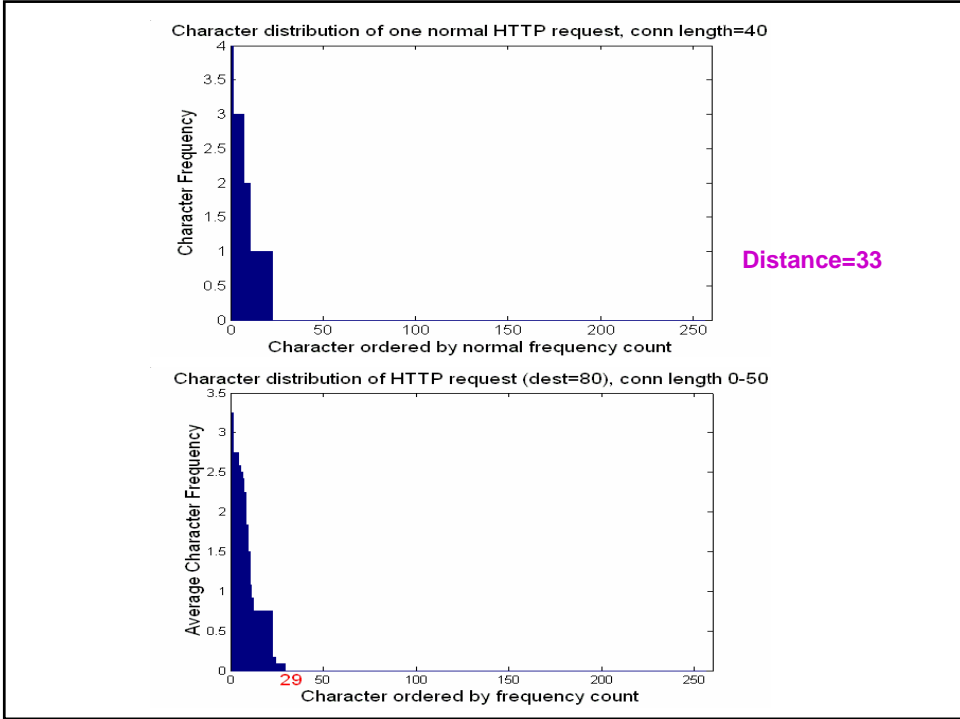




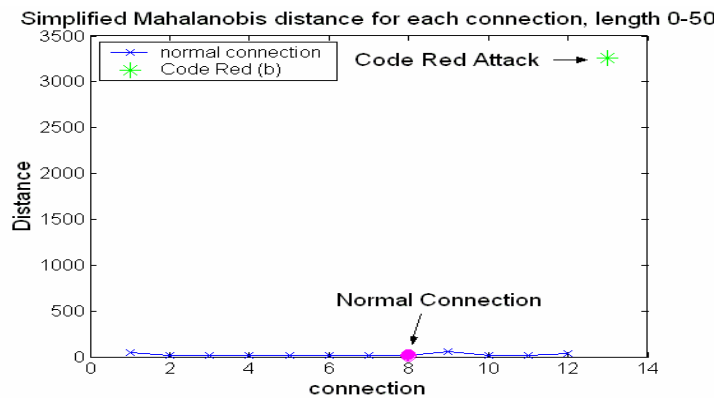
Test: **Code Red (a)**

Simplified Mahalanobis distance for each connection, length 380-385





Test: Code Red (b)



PAYL Discussion

- Efficient/bounded computation of 1-gram distributions
 - Line speeds achievable if we can detect anomalous packets
- Integration with firewall, or LAN-based filter possible (easy)
- Mimicry attack hard to implement
 - Attacker must know the distribution of the target payload stream, and have enough “budget” to embed attack mimicking the target distribution
- Robustness
 - Experiments ongoing to test
 - Integration/Correlation with other sensors and AD's to mitigate false positives
- Ongoing research
 - Automatic stream partitioning by length
 - Clustering of training sets for multiple centroids
 - Integration with Worminator distribution

Putting it together...

■ The components

- Payload Anomaly Detector at the perimeter (or behind the gateway firewall)
- Firewall sitting in front of servers to filter traffic or re-route traffic
- LAN or network-based AD detecting unusual tcp traffic from/to servers
- Mirrored servers to receive suspect traffic for analysis and response

The Strategy to kill worms at the border...

- Anomalous Payload detected at the perimeter shunted or re-routed to a mirror server
 - Either the payload is deemed anomalous by the detector, or a worm payload is received by a worminator site that has verified the new worm payload
- Mirror site replies to request to verify sender (eg., "Requested page is unavailable, try again.")
- Mirror site may also sandbox the request and test for server failure
- "Verified" worm payload converted to "Zstring" and distributed to other worminator sites
- A successful worm penetration requires interior filters and cooperation...

The Strategy to kill worms internally propagating...

- Limit damage to a small number of hosts via filters and server restarts
- Components:
 - Host-based wrappers and detector may stop server misuses
 - Restart servers to wipe away the worm, but loose state
 - Alternatively, LAN-based detectors that detect anomalous server/server interactions
 - Antura Insider integrated with filtering firewall in front of servers
 - Automated “re-start” of servers deemed infected