OWASP
AppSec Asia

SANS Dshield Webhoneypot Project

Jason Lam
SANS Internet Storm Center

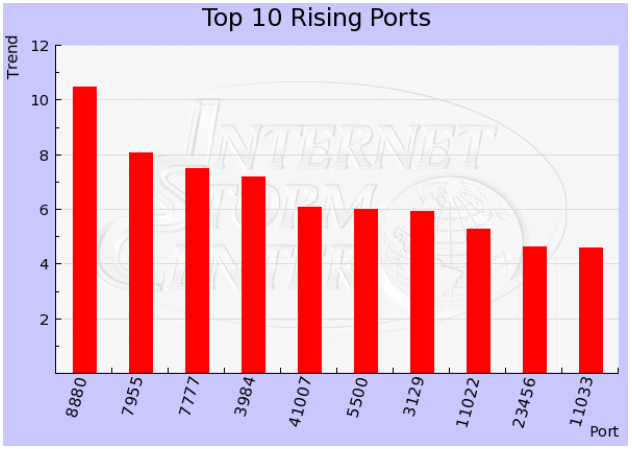
Introduction

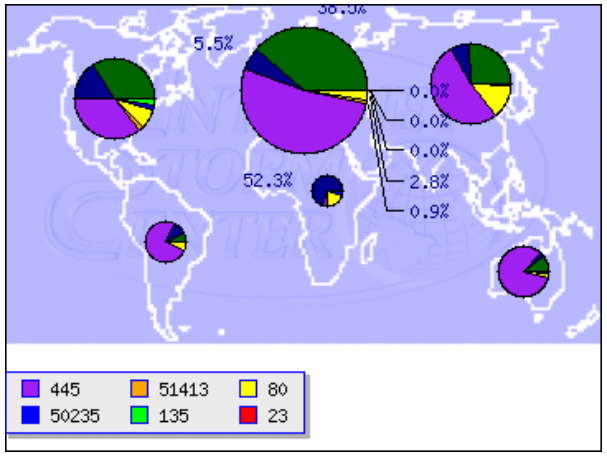
- Who is Jason Lam
- Agenda
  - Intro to honeypot design
  - Look at some data we collected
  - Future plans
  - Solicit participation

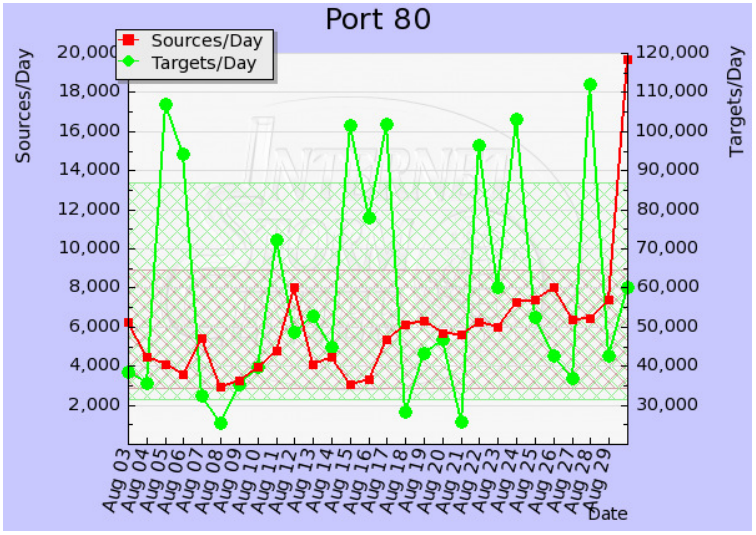
Dshield History

- Started in November 2000
- Collect perimeter logs (infrastructure)
- Provide attack trending information for community
- Notify community of upcoming attacks based on captured data

Dshield Screen Capture







Top 10 Reports

Top 10 Ports

by Reports		by Targets		by Sources	
Port	Reports	Port	Targets	Port	Sources
80	94907	80	68305	445	19167
4899	73910	4899	67908	51413	909
445	69028	1434	23718	80	848
1433	46402	445	5838	50235	544
1434	30401	1433	5196	135	439
139	21824	139	2264	23	420
443	11682	1080	2034	123	391
389	11242	22	1969	1433	389
135	7819	135	1860	25	337
161	7682	137	1541	48074	326

port report

Top 10 Source IPs

IP Address	Reports	Attacks	First Seen	Last Seen
058.243.161.051	1,396,158	152,277	2009-07-21	2009-09-02
218.204.137.156	974,291	150,872	2009-08-05	2009-09-01
202.101.180.165	1,049,184	148,693	2009-07-26	2009-09-01
218.075.199.050	549,698	137,902	2008-06-30	2009-09-01
202.107.196.099	564,862	133,145	2009-08-17	2009-09-01
060.161.078.144	334,407	125,202	2009-08-22	2009-09-01
218.030.022.082	487,091	109,109	2009-08-14	2009-09-01
218.023.037.051	780,584	101,341	2009-03-02	2009-09-01
058.057.017.194	145,760	99,602	2009-02-04	2009-09-02
124.173.184.018	140,949	97,700	2009-04-08	2009-09-02

Top Sources

Goal of Webhoneypot

Collect quantitative data about the prevalence of large scale automated attacks

- Capture logs from distributed honeypots
- Easy to deploy (by volunteers)
- Resilient against attacks
- Low maintenance
- Gather High definition logs
- Sufficient submitter privacy

Current Statistics

- 20 – 25 active sensors run by volunteers
- Collecting data since Jan 2009
- Averages about 18,000 requests/day
- 300-500 unique sources (IP) per day

Web Server Logs Not Enough?

- Web server logs does not contain the detail level
- Privacy issues with real logs
- HTTP Body
- HTTP Header

- Decision
  - Go with a software client, written in PHP
  - Collect entire request (header + body)
  - Software client developed by a group of volunteers

High – Low Interaction

- High Interaction
  - Maintenance nightmare
  - Get good insight into attack process
- Low Interaction
  - Less work, looks fake to the attacker
  - Less insight

- Decision
  - Ended up with low interaction design
  - Aim for fast scan detection, less for targeted attack

Serving Multiple Vulnerabilities

- Some mass exploit scripts look for specific page before sending attack
- Other script uses Google hacking to find target page

- Decision
  - Uses multiple template based on the request (REGEX)
  - Somewhat real looking page with graphics
  - Indexable (more on that later)

Regex Example

/PHPBB(.*)/i	101
/horde\-[\d\.]+\/(.*)/	1600
/(login_page.php$)/	1603
squirrelmail	1605

Log Collection

- Periodical
  - Store for period of time, then forward
  - More complicated
- Per request
  - As the request comes in, send to central server
  - Easy to implement
  - Similar to most logging protocol
- Decision
  - Push logs to centralized server at every request
  - Ignore “related” requests

Update Mechanism

- Templates need to be updated
- Honeypots should be refreshed periodically
- Not all honeypots should look the same

- Decision
  - Client updated daily
  - Clients separated in groups, get different set of templates

Images Requests

- Templates have internal reference
- Generate extra noise
- Some images or CSS requests are legit

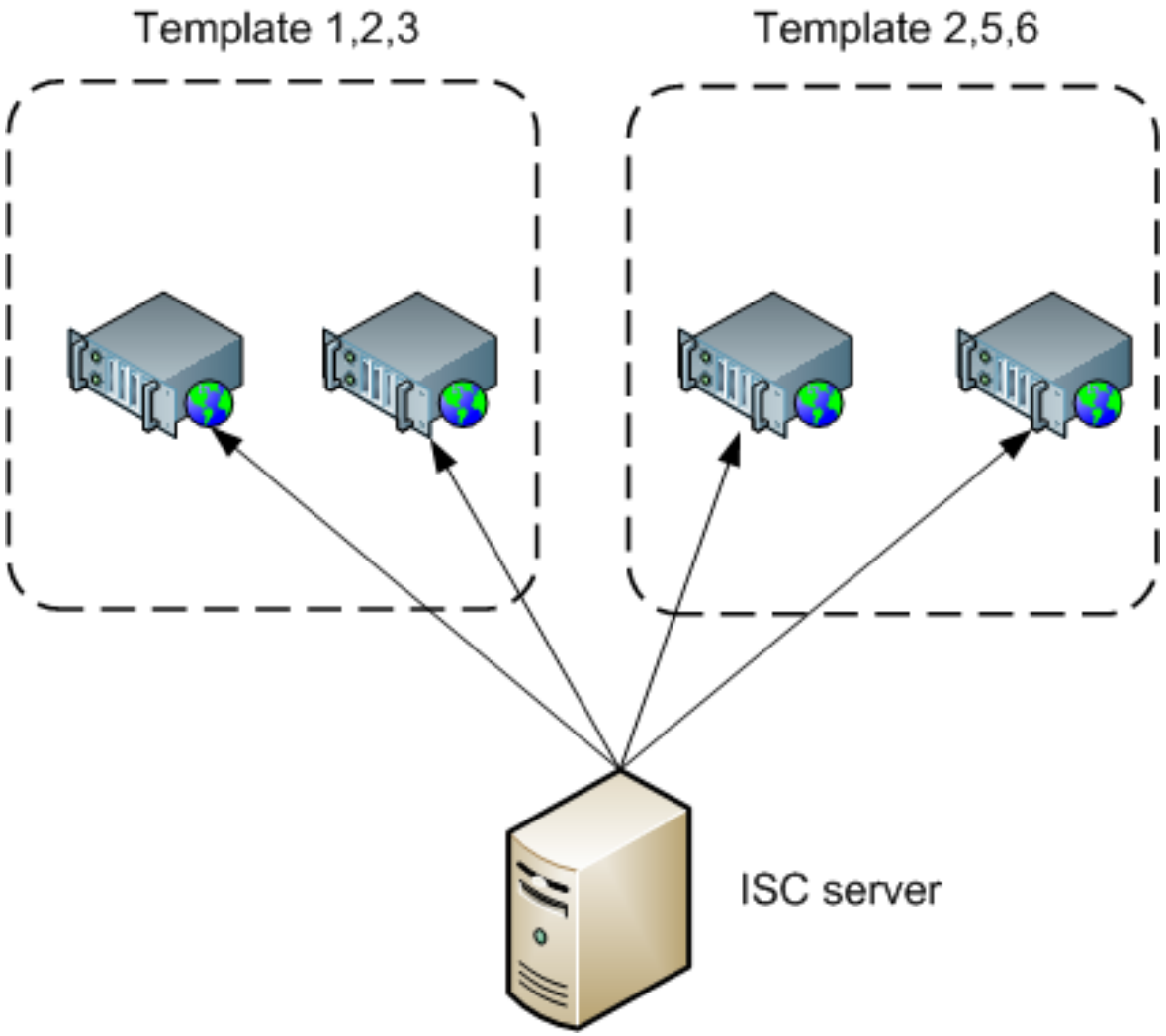
- Decision
  - Filter out all internal referenced images and CSS files
  - Might loss some details but overall cut down noise

Templates

- Serve multiple different version of pages according to the HTTP request
- Make the honeypot more "real"
- Uses regex
  - /phpbb/ -> template 1

- Use Google to link them together

Templates Linking (Search Engine)



Client Architecture - Request

- Processing requests (“index.php”)
  1. Request received by index.php
  2. Send complete request to collector
  3. Compared to regular expressions in configuration file
  4. Select template (or use default template)
  5. Add random link to template (to refer to other honeypots)
  6. Server template to scanner

Client Architecture - Update

- Update Process ("update.php")
  1. Script scheduled via cron
  2. Checks for new templates / new honeypot
  3. Uses nonce / hmac algorithm to validate update
  4. Downgrades will not be accepted, only upgrades

Client/Server

- Client Collects Requests
- Forwards request to server (“POST”)
- Server queues requests in flat files
- Cron job parses request and inserts them into the database

Server Side

- MySQL Backend
- PHP Frontend
- No expensive log management tools
- Uses the same backend architecture as Dshield
- Requests are parsed before they enter the database
- Some reports are generated on hourly basis

Server Database Architecture

- Daily tables for Requests, Headers and "Reports"
  - Request: Full requests
  - Headers: Parsed headers ("name" -> "value")
  - Reports: Summary including submitter info
  - Various "summary" tables which are updated whenever data is added to speed up web frontend.

Tables use "MyISAM" type to increase insert speed. Tables are not "updated" so transactions would be overkill. (can afford to loose a report once in a while)

Inserts are done in bulk on a schedule to avoid locking issue. ("DShield Architecture")

Other similar projects

- WASC Honeypot
  - Proxy based
- Glastopf
- Google Hack Honeypot (GHH)
- Honeyjax

Differences to Dshield Honeypot

- Dshield Honeypot is 100% open source
- Data available under “creative commons share alike license”
- Frontend updated “real time”
- Can use existing web servers and be placed in live networks

Outlet of Information

- Reports of information on the SANS Internet Storm Center website
- Limited public search capability (limited resources)
- Analysis done by public and ISC handlers

- Full information feed available upon request (researchers only)
  http://dshield.org/research.html

Basic Reporting (Demo)

- URL accessed
- Top 10 sources
- Headers with drill down
- Daily volume
- Limited header search

Attack Classification (Demo)

- Classify attacks into categories
  - RFI, SQL injection, XSS.....
- Regex on the request

What does the data look like

- Top Attacks
  - Remote File Include ~ 3000
  - Directory Traversal ~50
  - Proxy ~10
  - SQL Injection ~10
- RFI is by far the most common attack

Future Directions

- Community Driven!!!
  - Contribute templates
    - Validated before released
  - Attack classification, regex contribution
  - All these will be vetted by users (voting system)
- RFI
  - Actively analyze these RFI destination
  - Correlate the attacks to gather more intelligence
  - Collaborate with other groups

Want to Participate?

- Install Honeypot client
- Review our logs on ISC site daily
  - Alert ISC if anything is worth noticing
- Contribute templates
- Contribute regex for classification of attacks

Download Me Today

- Download from ISC portal
  - Sign in first and then look under My Information
- Requirement
  - PHP (if you want… work on a .Net version. Its not much code!)
  - Apache (Unix at this time)
  - Internet accessible

Questions

- http://twitter.com/jasonlam_sec
- jason@networksec.org

- Thanks!