Capture the Flag Games

Measuring Skill with Hacking Contests

By Riley “Caezar” Eller

Security Training Problems

- Computer security problems are here to stay
- Hackers know more than administrators
- Consultants are hard to measure
- Administrators need to minimize the delay between discovering a vulnerability and patching the problem

Solution: Security Games

- "Capture the Flag" or "CTF" games
  - Spread security techniques
  - Measure security skill
  - Strengthen technical and management skills
- CTF is difficult because
  - Security is arduous work
  - Player skill varies greatly
  - Scoring is often imbalanced or indeterminate

Spreading Security Techniques

- Accelerating the learning curve of our administrators is critical
- Convincing hackers to demonstrate their tricks in public is an exceptionally good way to do this
- Security professionals can take what they learn back to work

Measuring Security Skill

- It is difficult to measure the value of security consultants and employees
- "Capture the Flag" games have matured in the 21st century
- With a well-considered scoring system, these games can be used to accurately measure player skill

Learning From Performance

- Careful event logging allows us to measure
  - Effectiveness
  - Responsiveness
  - Team Coordination
- Learn more from competition than testing

Capture the Flag

- Originally a children's game to simulate small team combat, based on defending an immobile flag while trying to capture the flag of the other team
- Now, generally any game where teams defend and attack symmetrical targets simultaneously

DEFCON Game History

- Early years
  - Initially there were free terminals provided for conference attendees, who immediately tried to print text on the screens of other users

DEFCON Game History

- DEFCON 4
  - Formalized game with several servers provided as hacking targets
  - Judges were used to decide when a machine had been hacked and then awarded a point

DEFCON Game History

- DEFCON 5 and 6
  - Invited participants either to provide a server to be attacked, or to attack the provided servers
  - Problems abounded due to lack of oversight, poor scoring systems, unreliable networks and poorly configured target servers

DEFCON Game History

- DEFCON 7, 8 and 9
  - Game was dominated by the Ghetto Hackers
  - Chaotic blend of rule changes made the game unpredictable
  - During these years, winning was about hacking the contest rather than hacking the servers

DEFCON Game History

- DEFCON 7, 8, and 9
  - Other teams appeared to be competitive but Ghetto offense consistently overwhelmed the other teams
  - After DEFCON 9, we decided to produce a superior contest, meant to truly measure security prowess

DEFCON Game History

- DEFCON 10, 11 and 12
  - Contest provided by the Ghetto Hackers, based on a scoring system designed by myself
  - A few technical difficulties arose during DEFCON 11
  - With three years of engineering, the system is mature and stable, easy to customize and operate

System Topology

- Eight teams, arranged in a star pattern around a central network router
- One CAT5 cable each, connected to a custom router port
- Forward and reverse NAT hide all network addresses so teams cannot discriminate between each other and the scoring system

Scoring System

- A virtual 9th team resides in the router
- That machine runs in a loop, querying each of the other teams' computers
- These queries emulate normal usage, enabling many forms of attack
- Players abuse these patterns to gain unauthorized access to protected information

Tokens

- Periodically, the scoring system creates “tokens” for each service on the network
- It transfers them to each team and then waits for a random duration to pass
- While waiting, the system monitors the network to detect “information leaks,” which imply offensive scoring

Token format

- Tokens are three-part
  - Expiration: time for the token to be in play
  - Base: the hash of a random number
  - Token: concatenation of the base plus a DSA signature of the base
- Base-64 encoded for easy cut-and-paste

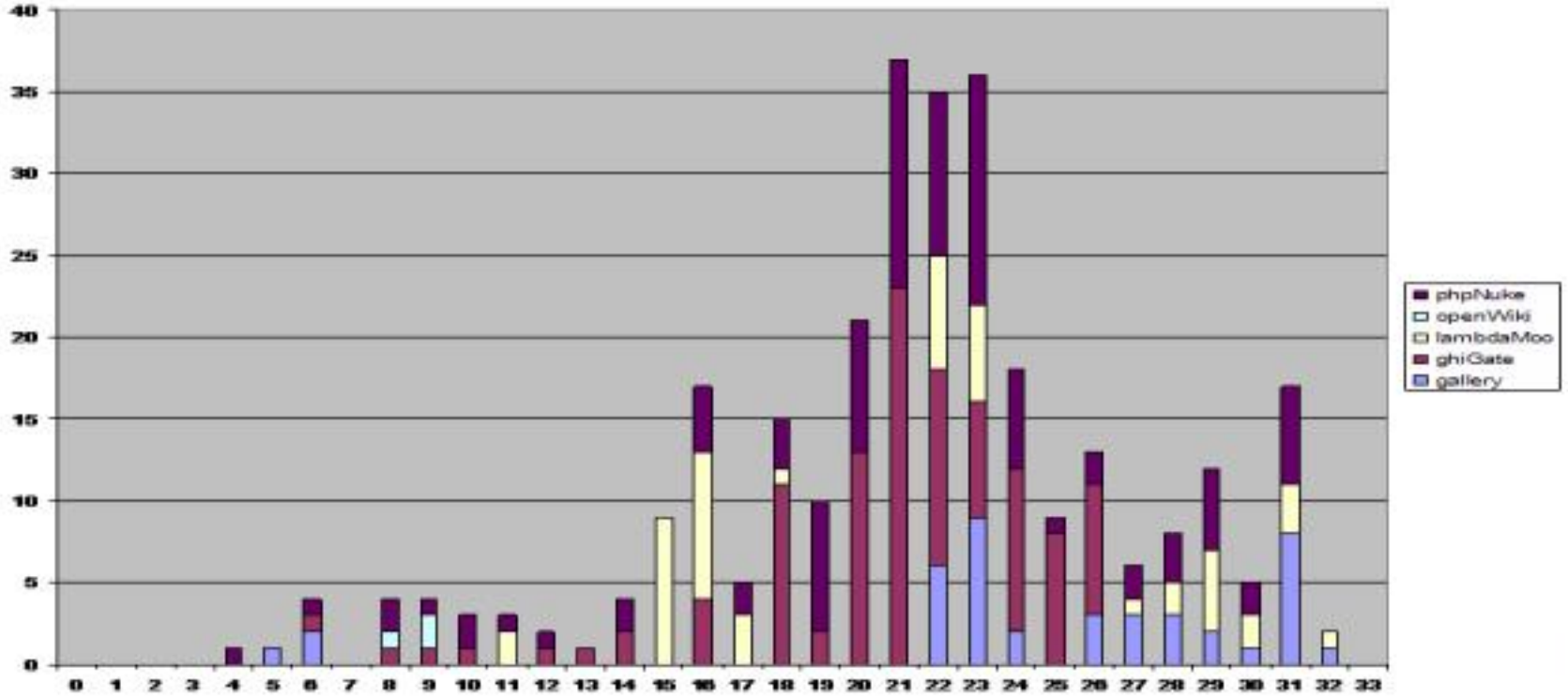
Scoring Points

- Periodically, the scoring system performs a “deposit” operation on each team’s services
- Each deposit is timed for “withdrawal”
- If another team penetrates system defenses to view a token, it can “exchange” the token using an upload script

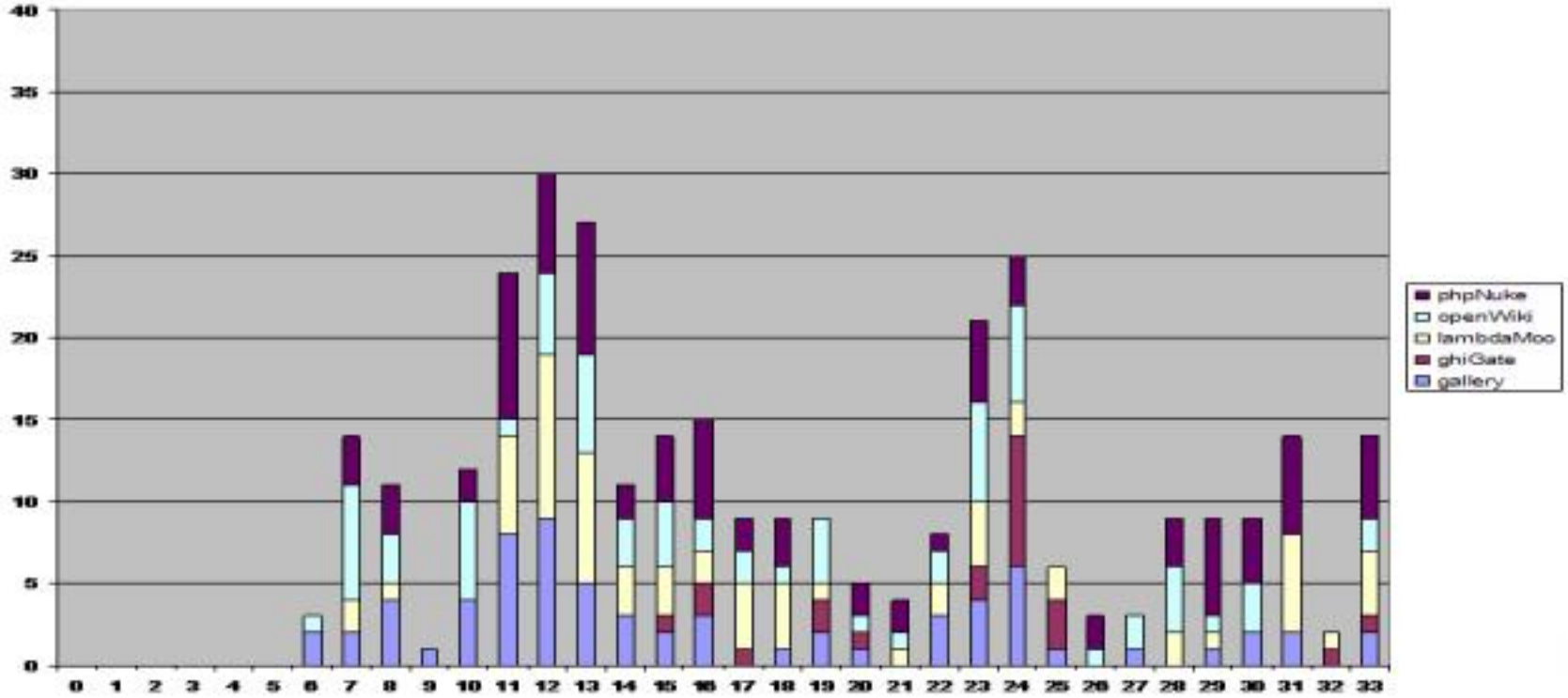
Scoring Points

- Each token is scored as
  - 1 offensive point for the last team to exchange it when the token expires; OR
  - 1 defensive point for the owner if no team exchanges it; OR
  - 0 points if the owner loses the information by rebooting or otherwise discards it

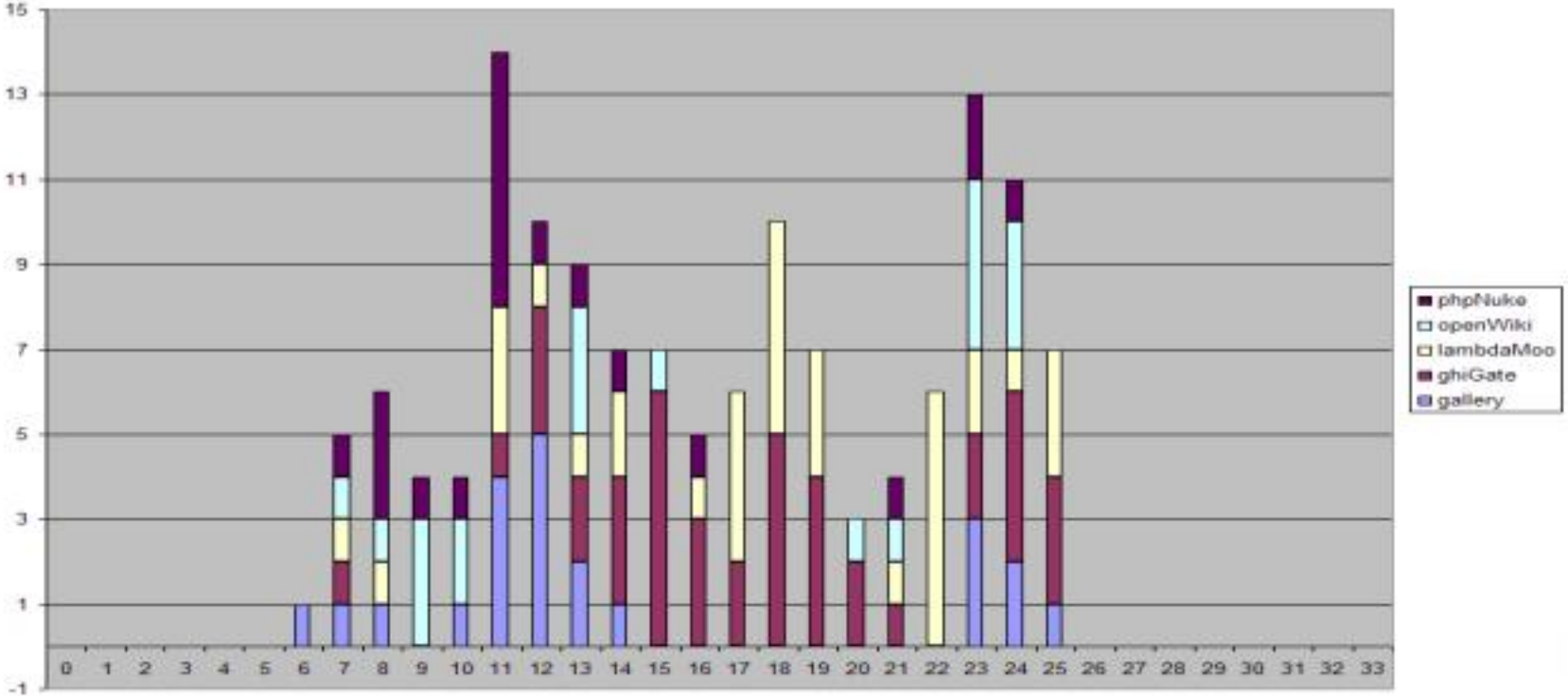
Offense (Yellow Team)



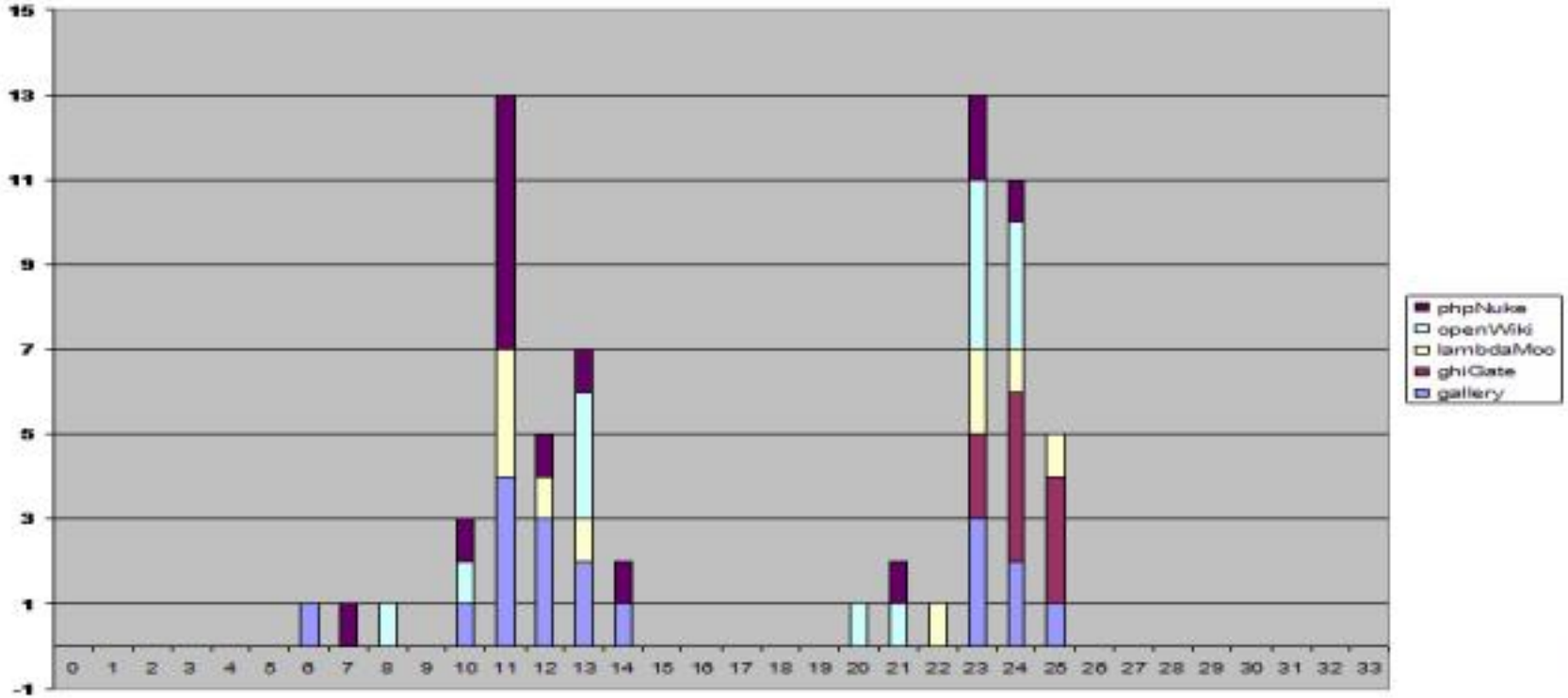
Offense (Red Team)



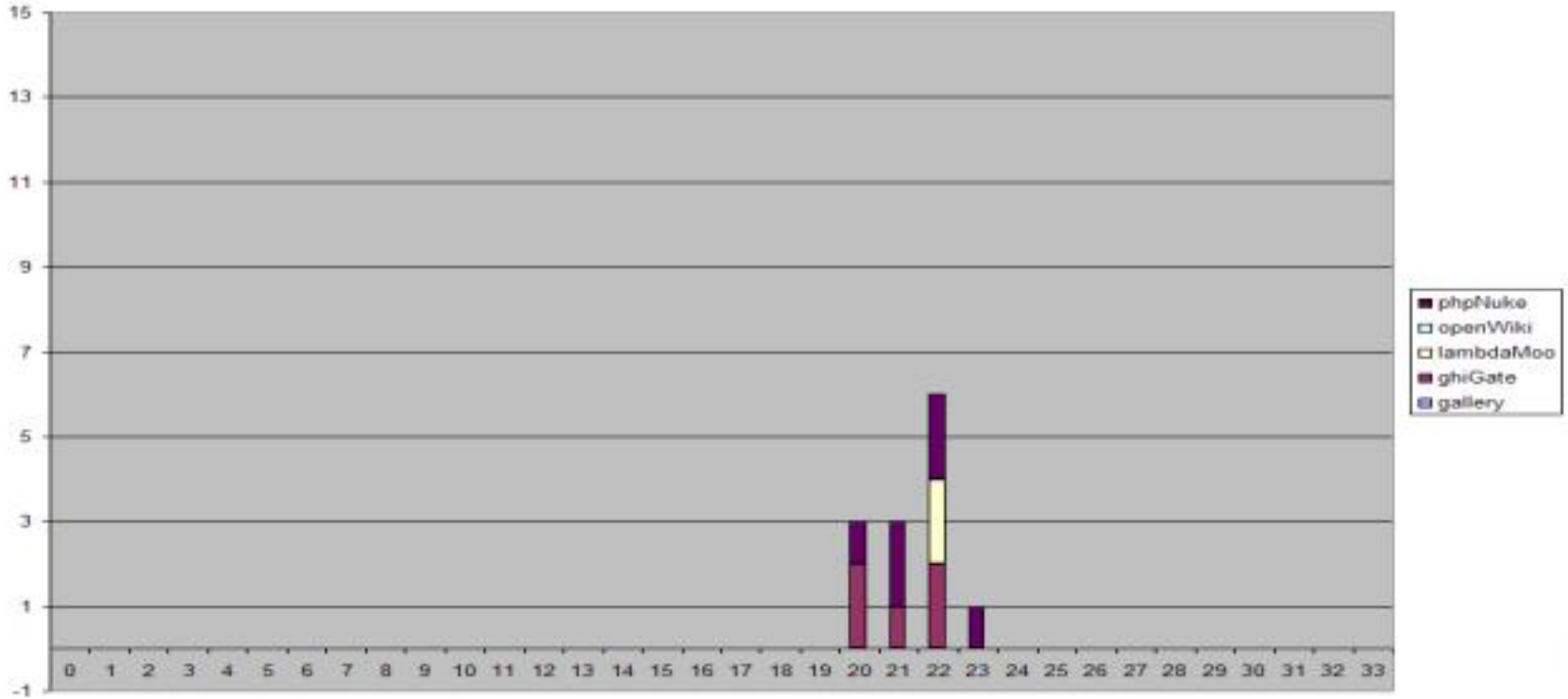
Total Attacks Against Yellow



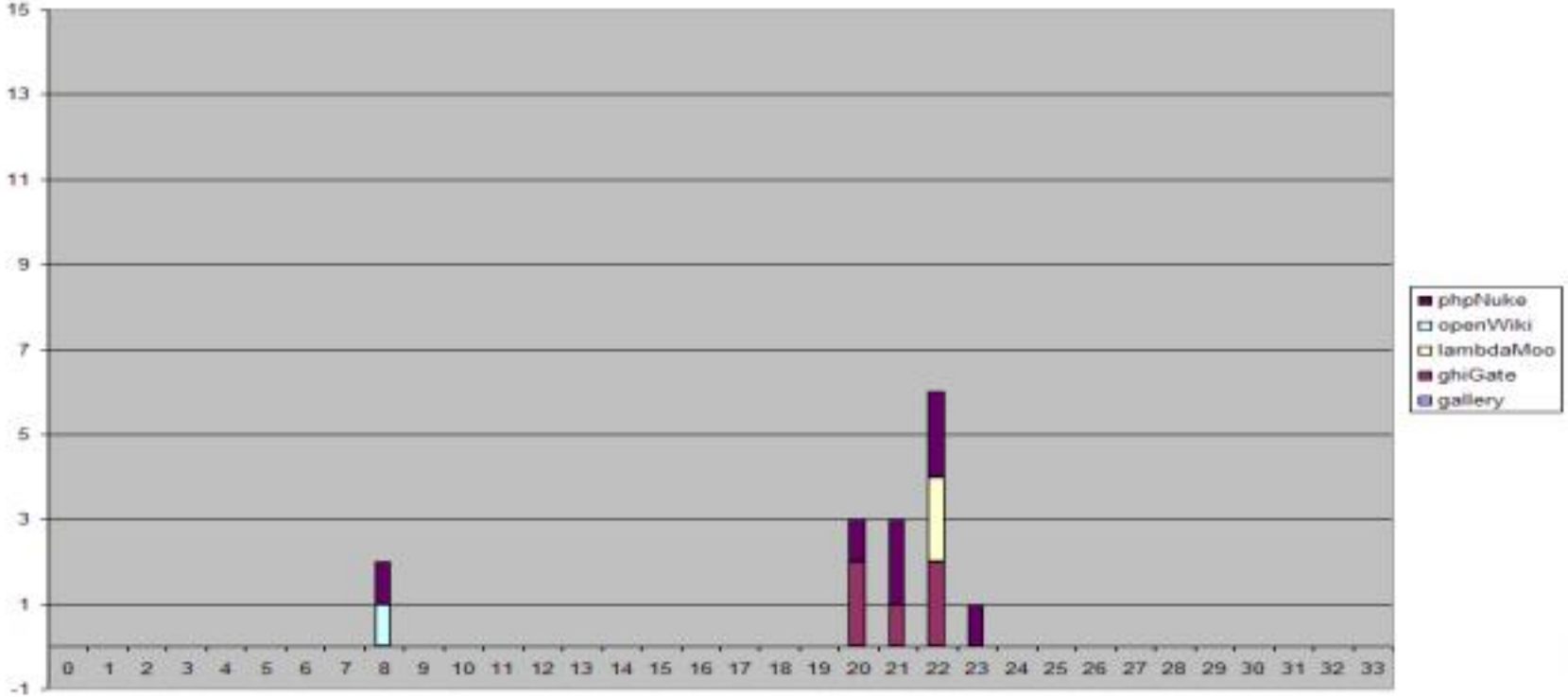
Red Attacks Yellow



Yellow Attacks Red



Total Attacks Against Red



What We Learned

- The scoring system determines the quality of the game
  - Small, simple scoring events are much more accurate than larger, more complex ones
  - Defensive scoring helps motivate teams early in the contest
  - Bandwidth penalties eliminate the need for many rules

Scoring Systems

- Terminology
  - Score: a simple event count
  - Ranking: a measure of players by direct comparison
    - Example: If A beats B, then A ranks above B
  - Ladder: the list of players, sorted by ranking
  - Rating: the scale of difference in ranking
    - Example: A is twice as good as B

Successful Rating

- Criteria for a successful rating algorithm:
  - Reproducible
  - Comprehensive
  - Detailed
  - Well balanced

Scoring Events

- As discussed earlier, the Root Fu scoring system measures the flow of information through a hostile network
- This provides reproducible, comprehensive and detailed measure of the following performance indicators:
  - Information theft
  - Durable defense
  - Service reliability

Balancing Scores

- The final requirement is to reliably estimate the value of a scored event
- To achieve this, we predict that the outcome will follow the compared ratings of two players
  - Example: If A has twice the rating of B, then we expect A to win and will not award many points if that happens

Rating Algorithm

- Initialize any new player's rating to 0.0
- Repeat:
  - Sample results from actual game
  - Predict outcomes based on current ratings
  - Compute the value of each event
  - Reduce the loser's rating and increase the winner's rating by the value of the event

Mathematics

- The Fermi function provides an excellent model for scaling numbers
- We must select two constants to begin:
  - S: the "speed" of the scoring
  - T: the "stability" of the scoring
    - 0.0 .. 2.0 : rapid adjustment, longer settling time
    - 2.0 .. ∞ : slower, smoother adjustment

The Formula

```
void Fermi( double& rating1,
  double& rating2, double result, double
  S, double T )
{
   double delta = rating2 - rating1;
   double expected = 1.0/(1.0+exp(-delta/T));
   double value = S * (result - expected);
   rating1 += value;
   rating2 -= value;
}
```

Discrimination

- Over time, the skills of each player or team will become obvious by their ratings
- Higher values of S will increase the speed with which scores change, but surprising performances will damage the reliability of the ratings

Convergence Time

- After a few hundred scoring events we should have no problem determining the relative capabilities of players
- Reaching the balanced state where each player's ability is accurately reflected by the rating system is called "convergence"

Display Ratings

- There are two problems with the rating system so far
  - About half of all teams will have negative rating values, which incorrectly implies that they are “worse than zero”
  - The scores are continuous and may appear as small fractions

Display Ratings

- To solve this problem, we need to convert all the scores into a positive range and scale them to match human expectation
- One convenient formula is:
  - `S=1000.0`
  - `T=(best_score-worst_score)/2;`
  - `Display=S/(1.0+exp(-rating/T));`

Rating Confidence

- If we have no data—because a player has just joined, or because a player is simply not playing—then our best estimate is that the player has average strength
- The solution is to require participation in a fixed number of events before publishing a player's rating

Comparing Rating Systems

- Compared to Elo and the system used by the World Chess Federation (FIDE), this proposal
  - Offers easier-to-explain ratings
  - Is somewhat more resistant to erosion over time
  - Is more fine-grained

Conclusion

- We should encourage security games to improve the abilities of our network administrators
- Security games rely on the scoring system to provide meaningful results
- Scoring systems are well understood but complex, so we should adopt a consistent scoring and rating scheme

Thank You

Riley “Caezar” Eller

Director, Special Projects Division

CoCo Communications Corp