Quantitative Approaches to Software Security & Information Privacy

Rachel Greenstadt, David J. Malan, Stuart E. Schechter, Michael D. Smith
http://www.eecs.harvard.edu/securitas/

Goal: Develop security risk and strength metrics that enable us to optimize technical security investments in and develop new techniques for large and complex systems.

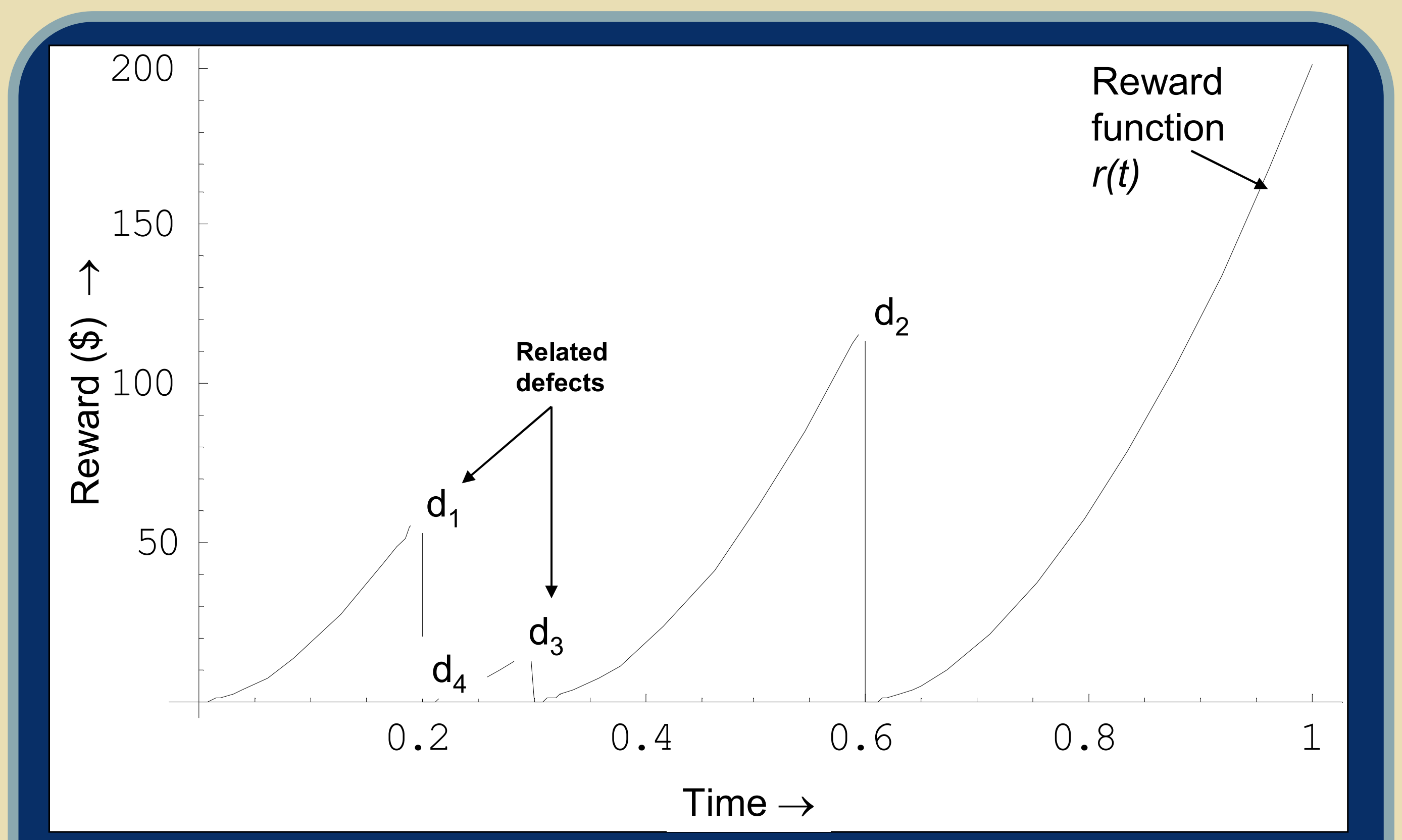
What level of security is required to successfully deter threats?

Economics of Software Security

The *Security Risk* faced by a system in running a software package is a function of

1) the number of potential adversaries,
2) the adversaries' incentive to attack,
3) the risk posed to the adversary of attacking the system,
4) the time, effort, and other resources required in a successful attack.

To increase (4) is to increase the *Security Strength* of the software. By determining the expected cost to find a vulnerability in a software system, we can begin to measure *Security Strength*. By offering rewards for vulnerability discovery, software vendors can measure the security strength of their software using markets.



A bidding strategy for purchasing related and unrelated vulnerability reports.

Today there exists a legitimate competitive market for vulnerability reports.

What are meaningful metrics for privacy loss in our modern lives?

Privacy in Constraint Optimization

Focus: Distributed constraint optimization (DCOP) for resource allocation, meeting scheduling, etc.

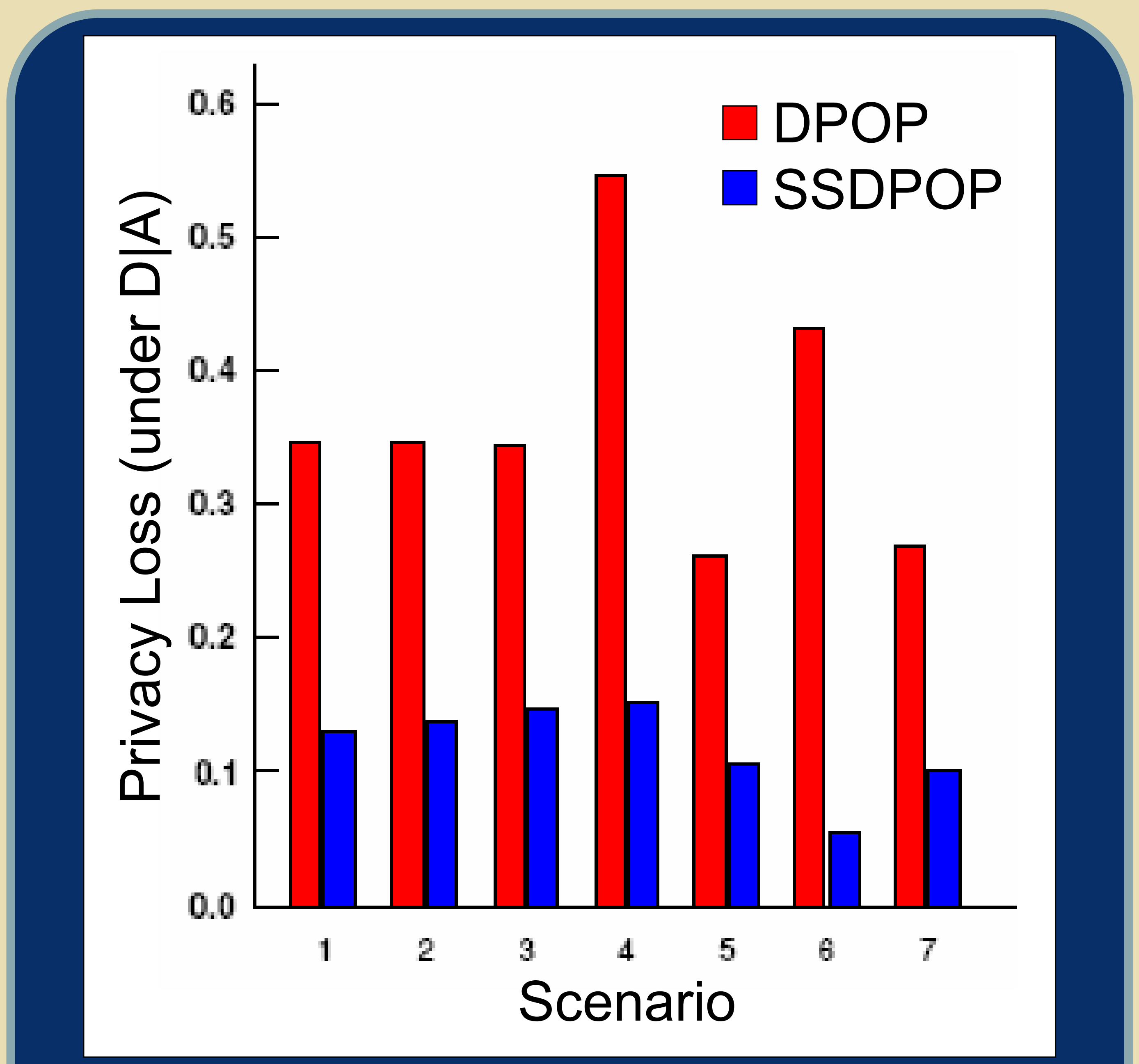
Issues:

- Distributed can be worse than centralized;
- Current metrics provide little insight.

Approach:

1) Develop metrics for measuring privacy loss;
2) Categorize sources of vulnerability;
3) Design new algorithms to increase privacy.

Contributions: Used secret sharing to eliminate the *initial vulnerability*, the worst of four vulnerabilities identified, in DPOP, shown to be one of the best DCOP algorithms. Assumed honest but curious adversaries. SSDPOP maintains structure of DPOP and adds little additional overhead.



Secret-sharing DPOP (SSDPOP) reduces unintended information flow.

Do computing configurations exist that are measurably "easier" to defend than attack?

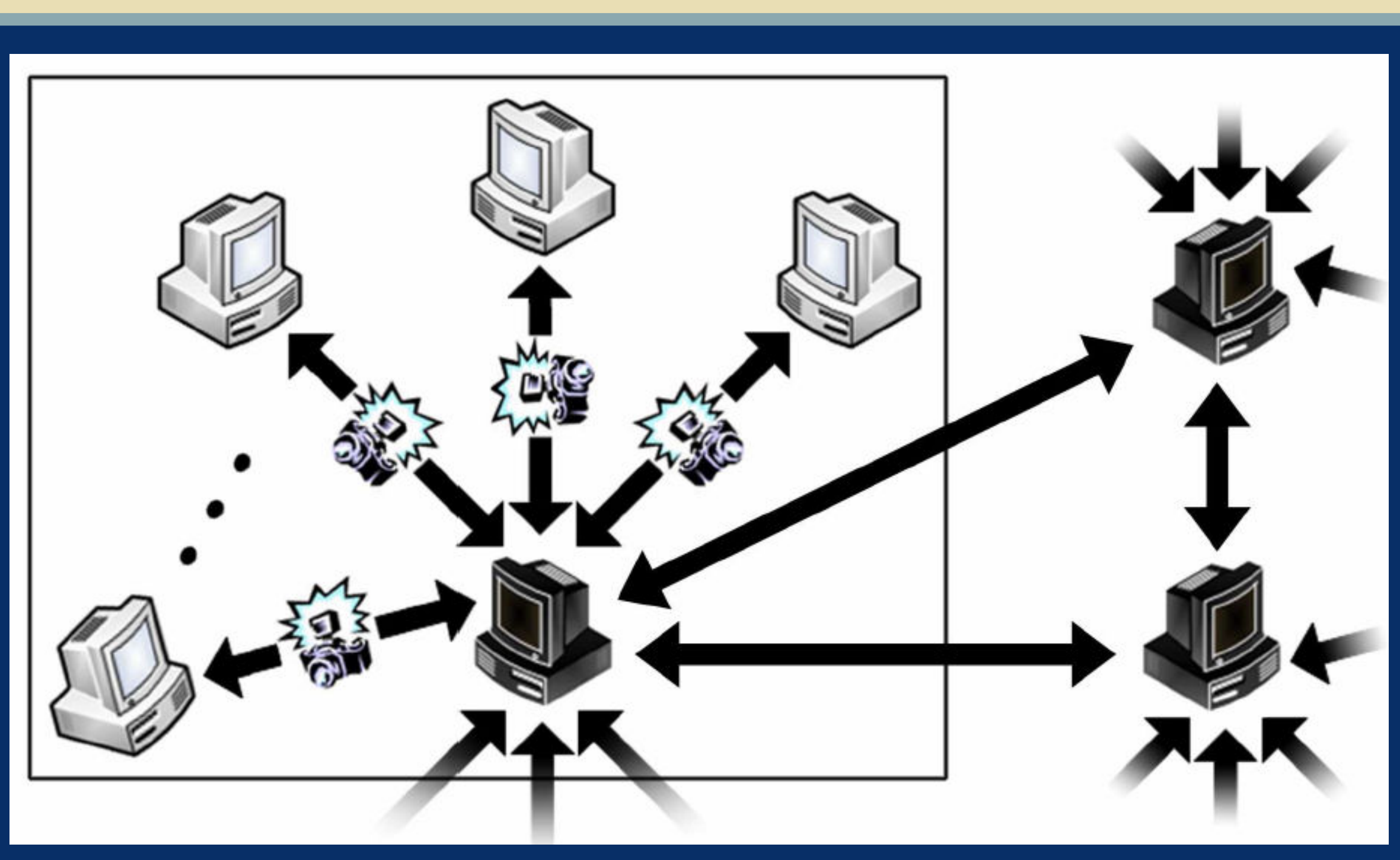
Avoiding False Positives in Host-Based Worm Detection

Behavior-based detection compares a host's current actions against its prior actions:

+ Can detect previously unseen worms;
– Often flags benign applications as worms or is easily circumvented by adversary.

A more robust definition of anomalous behavior: a host's behavior is anomalous if it correlates too well with other networked, but otherwise independent, hosts' behavior.

Proposed a distributed IDS for fast-propagating worms that searches for *temporal consistency* in system call snapshots to identify worms with near-zero false positives.



Monitored 10,776 processes, comprising 873 unique non-worms on 30 Windows hosts. Only 14 of the 873 non-worms ever appear similar to real worms.

NSF Cyber Trust Principal Investigators Meeting
January 28-30, 2007
Atlanta, Georgia