Towards Quantitative Security Evaluation

Yves Deswarte,
Mohamed Kaâniche

LAAS-CNRS
Toulouse, France
Fault forecasting

= Evaluation:
  - Gain confidence that system dependability is satisfactory
  - Select architecture/components to achieve the best dependability-performance-cost trade-off

❖ Quantitative measures
  - Reliability: \( MTFF = \text{mean time to first failure}, \)
    \[ R(t) = \text{prob}_{\text{continuous service}}(t) \]
  - Availability: \( \text{MTBF}/(\text{MTBF+MTTR}) \),
    \[ A(t) = \text{prob}_{\text{correct service provided when needed}}(t) \]
Basic assumption

- Faults = elementary component failures (or other rare physical phenomena)
  
  Model = independent stochastic processes with known distributions

- OK for physical H/W faults and most environmental faults

- ~OK for most S/W design faults (bugs)

- Not OK for attacks or malicious design faults
Security Evaluation

- Usual techniques
  - Evaluation criteria (TCSEC, ITSEC, CC, ...):
    - Qualitative evaluation
  - Risk assessment: Subjective evaluation of vulnerabilities, threats, consequences
    - These are static analyses rather than dynamic: "How the system has been built?" rather than "How is it operated?"
Quantitative security evaluation

- Measure = effort needed for a possible attacker to defeat the security policy

- Objectives:
  - Take into account security/usability trade-offs
  - Monitor security evolutions according to configuration and use changes
  - Identify the best security improvement for the least usability change
ESOPE: General approach

- Identify security objectives: security policy

- Model (operational) system vulnerabilities

- Model the attack processes

- Compute significant measures
Vulnerability modeling

Privilege graph

- **Node** = a set of privileges (user, group, role, ...)
- **Arc** = a method to transfer privileges = vulnerability
- **Path** = set of vulnerabilities usable by a possible attacker to reach a target
- **Weight** = for each arc, effort to exploit the arc's vulnerability

1) X can guess Y's password
2) X can install a Trojan horse that Y can activate
3) X can exploit a flaw in Y's mailer
4) Y is a subset of X
5) Y uses a program that X can modify
6) X can modify a "s-uid" program owned by Y
7) X is in Y's .rhosts
Assumptions on the attack process

- Attack process = all possible successful attack scenarios

- Reasonable assumptions
  - The attacker knows only the vulnerabilities that can be exploited with the privileges he already owns.
  - The attacker will not exploit vulnerabilities which would give him privileges he already owns.

- Plus one out of the two following assumptions:
  - *Total Memory (TM)*: the attacker remembers all the vulnerabilities he did not exploited in the previous steps, and he can "back-track".
  - *Memory-Less (ML)*: the attacker considers only the vulnerabilities that can be exploited with the new privileges he just acquired.
Attack Process Examples

ML Assumption

TM Assumption
Measure computation

① Identify the attacker-target couples

② For each couple, compute:

**METF-ML**: Mean Effort To security Failure (i.e. to reach the target) with ML assumption.

**METF-TM**: Mean Effort To security Failure with TM assumption.

**Shortest Path**: Mean effort to go through the shortest path.

**Number of Paths**: Number of possible paths from the attacker to the target nodes.
ESOPE Tool Set
(Evaluation de la Sécurité OPERationnelle)
Objectives:

- **Validate the approach:**
  - Assess the measure pertinence wrt. system changes (configuration, users, ...)
  - Feasibility of a full-size system evaluation.

- **Was not aimed:**
  - Correct the identified vulnerabilities
Experiment context

Target system:
- Unix
- 700 users - 300 machines - LAN
- 13 months (June 1995 - July 1996)

13 types of vulnerabilities (files .rhosts, .*rc, passwords, etc.)

Security objectives:

<table>
<thead>
<tr>
<th>Attacker</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objective 1</td>
<td>insider root</td>
</tr>
<tr>
<td>Objective 2</td>
<td>insider admin_group</td>
</tr>
</tbody>
</table>

4 difficulty levels:

<table>
<thead>
<tr>
<th>Type</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>immediate</td>
<td>10</td>
</tr>
<tr>
<td>easy</td>
<td>$10^2$</td>
</tr>
<tr>
<td>difficult</td>
<td>$10^3$</td>
</tr>
<tr>
<td>very difficult</td>
<td>$10^4$</td>
</tr>
</tbody>
</table>
Results (1)

insider → root

Number of paths

- METF-ML
- METF-TM
- METF-SP

Dates: 06/95 to 07/96
Results (2)

insider → admin_group

Number of paths
- METF-TM
- METF-ML
- Shortest Path

data points for years from 06/95 to 07/96.
Comparison between measures

- The shortest path (SP) is not sensitive enough to identify important events.

- The number of paths (NP) changes too often and would produce a large number of false alarms.

- METF-ML presents a good sensitivity to important events.

- METF-TM is easier to interpret, but is sometimes too complex to be computed.
Problems

- Is the model valid in the real world?

- TM and ML are 2 extreme attacker behaviors, but what would be a "real" attacker behavior?

- Weight parameters are assessed arbitrarily (subjective?)

- Tenacity? Collusion? Attack rates?

→ We need real data!
Validation based on real attack data

- Collect real life data to learn and analyze attackers behaviors, tools and tactics

- Objectives
  - Validate attack assumptions
  - Analyze adequacy of the privilege graph to describe new vulnerabilities and derive attack scenarios
  - Extend security evaluation approach by taking into account distribution of attacks in time, correlation between attacks, etc.
Honeypots and Honeynets

- **Honeypot**
  - A security resource whose value lies in being probed, attacked or compromised
  - Anything going to or from a honeypot is likely a probe, attack or compromise

- **Honeynet**
  - A network of honeypots
  - All systems placed within the Honeynet are production systems: Solaris, Windows, Linux

http://www.honeynet.org/alliance/
Example of Honeynet
The threat is real!

- Computers scanned dozens of times a day
- Fastest time a honeypot manually compromised: 15 minutes (automatic, 92 seconds)
- Time before a default Linux Red Hat 6.2 successfully hacked is 72 hours
- 100% - 900% increase of activity from 2000 to 2001
- It's only getting worse

http://www.honeynet.org/papers/stats
Perspectives

- **Data collection**
  - Several honeynets (different domains, locations, etc.)
  - Need to analyze if data collected from different locations (e.g., .com vs. .edu) exhibit similar or different statistical patterns

- **Data Analysis**
  - Identify attacks and characterize their distribution in space and time
    - known and new vulnerabilities
    - attack scenarios
    - trend analysis

- **Security modeling and evaluation**
  - Take into account the lessons learnt from data
  - Analyze how results are useful for designers/administrators
What we do NOT expect:

- Plausible attack rates / effort distribution

... necessary for "reliability / availability" measures
References


