Defense by Deception
SaTC 2022 PI Meeting Breakout Group Report
Lead: Mark Grechanik
Scribe: Susan McGregor

Problem Domain/Summary

Defining deception

- Agents choose actions to manipulate the beliefs of other agents to take advantage of their erroneous inferences.
- Deception is a con
- Need to hide that you're planning to deceive
- Different strategies with respect to short- vs. long-term cons
- Deceiving is lying while cheating is breaking the rules (point being that in attacking there are no rules)
- Goal of deception is to increase cost of attack, but ideally also to learn information about the attacker

Example: Attackers write viruses that pose as legitimate programs, legitimate inquiries (phishing) etc. As a result, prevailing beliefs are changed (or unchanged) by the deception (e.g. in phishing, recipients assume legitimacy of emails in general and are reluctant to change that).

Traditional approaches

Malicious applications deceive us into believing we will receive legitimate services/benefits. Hiding malicious intent and masking malicious action is common

Hiding (Dissimulation)

- **Masking** by blending with background info (e.g. obfuscation)
- **Repackaging** allows malicious agents to hide the malicious code within/alongside a legitimate app
- **Dazzling** by adding noise that doesn't inhibit desired performance

Showing (Simulation)

- **Mimicking** creates partial functional equivalents for the protected target that run in parallel to confound agents
- **Illusions** present new fake features to attract attackers away from real targets (e.g. honeypots)
- **Decoying** creates fake resources that include beacons to send a signal when a decoy has been accessed (e.g. files with attached alarms)

Key Challenges

Bespoke DbD (Defense by Deception) efforts usually:

- Hide actual functionality/approach of the system
- Create false beliefs for attackers
- Trick attackers into quickly revealing themselves and/or abandoning the attack

How can we automate the design of DbD tools that work on generic systems? For example:

- Systems that automate distortion of the perceived reality of the system with respect to attackers

Obstacles

- Possibly reduced performance
- Deception strategies are difficult to design

e.g. SSL - actual encryption is not so strong, but it's hard enough to do in practice that it's not worth it to apply the attack arbitrarily

- *Is broad-based encryption a form of DbD?*

<u>Potential approaches/core ideas</u>

- An automated (scalable?) approach would be good
- Define steps that will lead to detection before harm is enacted
- Force the agent to perform actions based on the distorted beliefs -> Why?

Program generation

- Create artifacts that trick malicious applications into executing an attack in e.g. a detection sandbox rather than a real system
- Game-theoretic analyses using e.g. Monte-Carlo simulation

Example:
Mitigating DDoS w/DbD

- Attacker aims to exhaust network with trash packets
- Create appearance of vulnerabilities to attract attackers, and know to actually trash those packets

Solution steps:

- Model system, resources and possible types of attacks
- Synthesize DbD strategies to the resources and assign defense costs
- Generate game theoretic moel and simulate the DbD to simulate what type of defense is most successful

Short-term target: Mobile apps
Long-term target: Automate design of tools for securing commercial software/applications

<u>Other perspectives</u>

Privacy applications of deception:

- "Social camouflage" via pseudonyms in online gaming platforms
- Obfuscation - cockney rhyming slang
- Deception for defense - avatars in VR/AR

Is part of the problem that there is just asymmetry in the way that different parties value these resources? As in, a lot of the technology is doing something that people just don't care enough to secure?

Why limit to devops when so many attacks target humans?

Is game theory used in the real world (as we know that honeypots are)?

- As in, can it be used to design systems?
- Used extensively in the financial system to estimate performance of various funds
- Are they used to determine whether certain strategies are effective in actual software development? We don't know.
- Do we have the data about attacks needed to do this? Not really.
- Simulations are still expensive to design and run.

Some software does code obfuscation - can this be done for defense?

Also, environment is always changing so there are ongoing costs to both sides.

Some deception will work, some may not - what are the feedback mechanisms? Game-theory vs. agent-based simulation? How does this model accommodate change over time?

Key value is the budget for each agent

- Defender may incorrectly model the budget of the attacker
- What if we lack decent information about the budget?

Maybe start with data/system that needs protection and then design the DbD con that will add cost to an attack

Hardware
Can we apply moving target to hardware where there are more limited resources? Is it too expensive to allocate e.g. RAM to specific operations?

Deception-based crimeware is a decentralized industry, but defense is centralized. Can decentralization of defense (and maybe resources in general?) help?

A lot of the data/deception tools have data without clear intelligence coming out of them
A key challenge is to take the data that is coming out of the deception attack and learn about the attacker