Game Theoretic Malware Detection

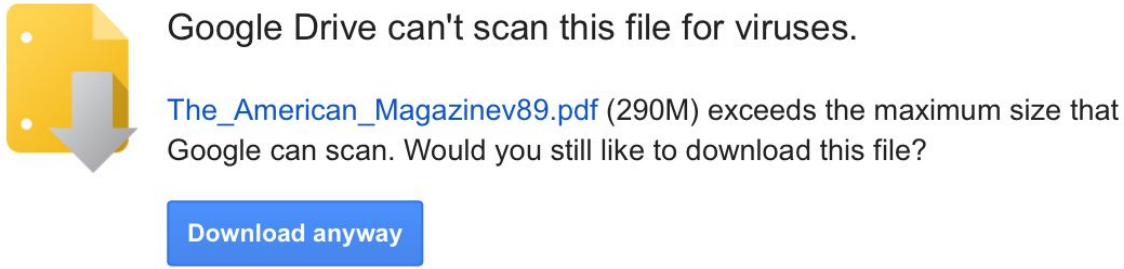
Revan MacQueen, Nicholas Bombardieri,
Karim Ali, James R. Wright
University of Alberta

The Problem

- Platforms like Gmail want to keep users safe from malware.
- To do this, files are scanned for malware.

- This helps to keep users safe.
- But there is a catch...

The Problem



- Not even Google is safe from resource constraints!
- This resource constraint is caused by:
  - The large number of files shared every day.
  - Scans taking longer depending on file size.

The Problem

How to lower costs and avoid creating weaknesses in our system?

- Pick a few overall strong tools?
  - Attacker can learn what tools and analyses you're using.
- Pick only the best tools based on which attacks are likely to be made.
  - An attacker could change their actions to exploit this prediction.
- So what should we do?

This Problem in Another Context

- Previous work has considered airport security.
- Security needs to defend airport from potential attacks.
- There are far too many ways an attack could be carried out to guard them all.
  - Can’t guard EVERY possible way in.
  - Can’t frisk/scan EVERY person coming into the airport.
- The best solutions in terms of security are also the most costly.
- So what’s the airport security to do?

Software Assistants for Randomized Patrol Planning for The LAX Airport Police and The Federal Air Marshals Service

Manish Jain, Jason Tsai, James Pita, Christopher Kiekintveld, Shyamsunder Rathi, Fernando Ordóñez, Milind Tambe

Airport Security

- If they want to save resources, security could think of ways to reduce coverage:
  - Could simply guard fewer entry points?
  - Develop some rules for which people to search?
  - Regular patrols?
- So what's the problem here?
  - Attacker will observe what security does and exploit any holes in a policy.
- What's the solution?
  - Intelligent randomization.
  - The attacker will not be able to predict exactly what security will do.
  - Takes care of resource constraint issue.
- But what is the best way to randomize?

Security Games

How do we find the optimal randomization? With a model!

- We can model security domains as Stackelberg security games, a concept from the field of game theory.
- These games involve two players, a defender and attacker.
  - These players act sequentially; defender first, attacker second.
- Both players take actions in order to maximize their utility.
  - But players must act strategically.
    - Attacker considers defender’s strategy
    - Defender must consider what an attacker would rationally do in response to their actions
    - i.e. they *best respond*

Stackelberg Security Games

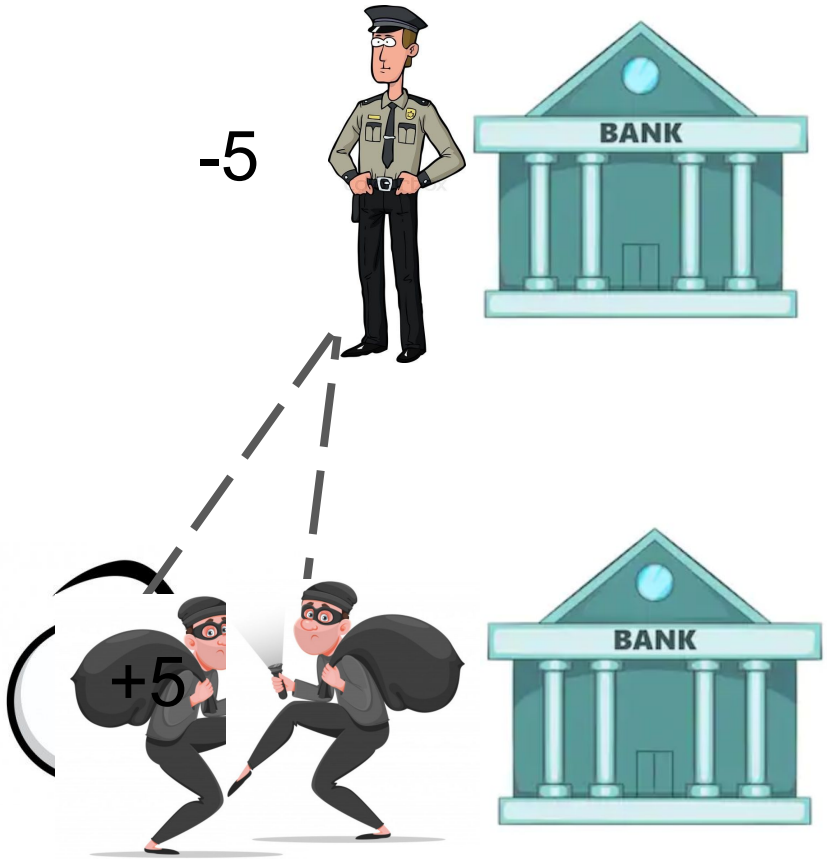
1. Defender chooses a strategy.

2. Attacker observes defender's strategy for as long as they want.

3. Attacker chooses an action to take.

4. Outcome is determined.

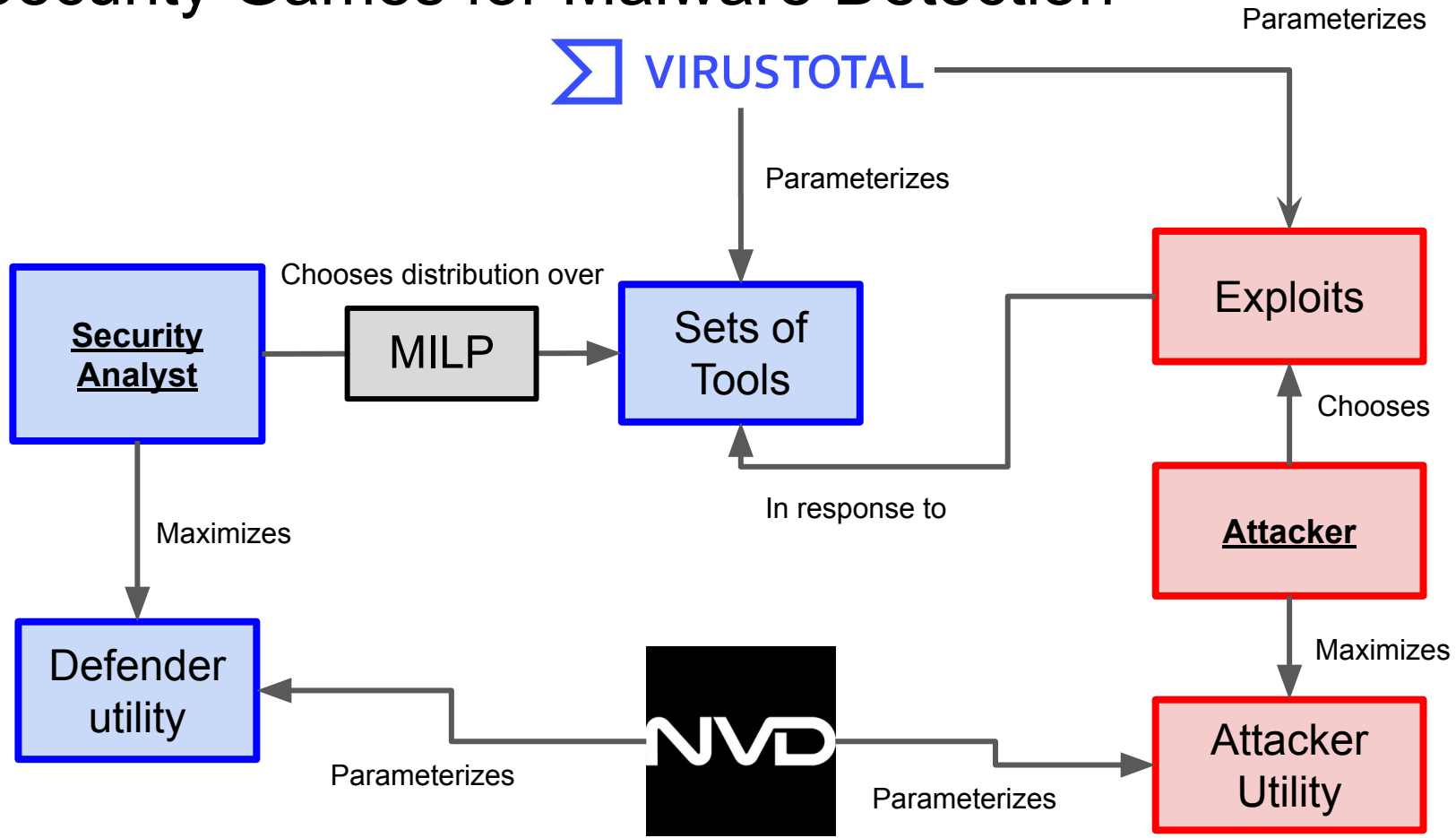
5. Players get utility based off of the outcome.



Back To Malware Detection...

- We model the relationship between security analyst and malicious party as a security game.
- The defender: security analyst
  - Not practical to use all available malware detection tools.
  - Different tools use different scanning strategies with different results.
  - Overlap in capabilities of tools can lead to inefficient use of resources.
- The attacker: malicious party
  - We assume they are *strategic*, i.e. will change their actions to maximize their utility in response to the defenders actions.
    - For this reason, the defender cannot rely on tools that have performed well in the past.

Security Games for Malware Detection



How does our proposed method of optimal randomization compare to other approaches?

Baseline Defender Strategies

In order to evaluate optimal randomization, we need to choose some baseline strategies for choosing which sets of tools (or schedules) are used:

- **Uniform Randomization (uall):** Randomly pick schedules.
- **Best Average Detection Rate (ba):** Defender always chooses the schedule with the highest overall detection rate.
- **Randomized Best Average Detection Rate (u10):** Same as (ba), but uniformly randomized over the ten schedules with the highest overall detection rates.

Baseline Defender Strategies

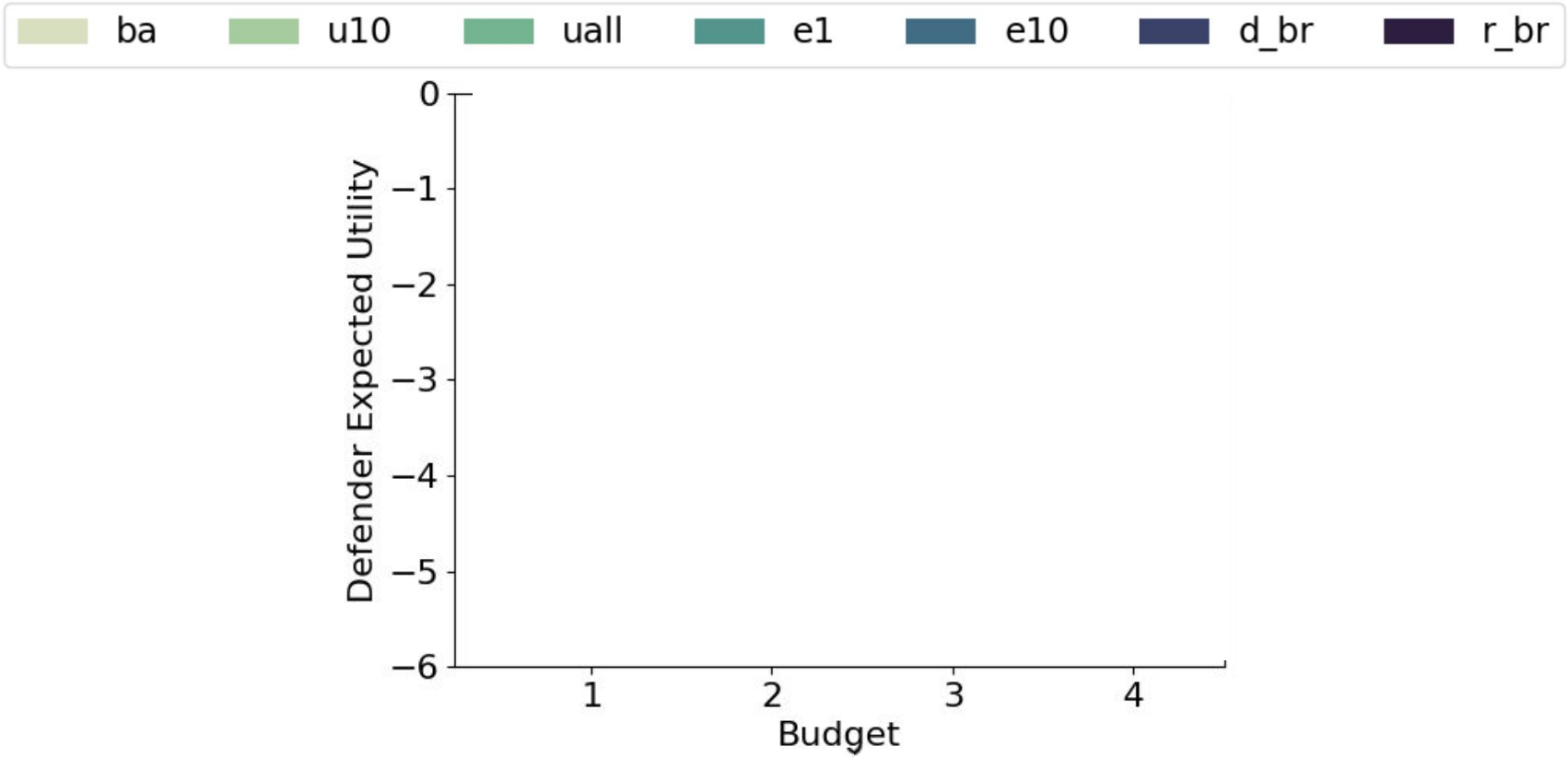
- **Highest Expected Utility (e1):** Defender chooses the schedule with the highest expected utility based off of a historical record of attacks.
  - This doesn't consider how a best responding attacker might react to this.

- **Randomized Highest Expected Utility (e10):** Same as (e1), but uniformly randomized over the ten best schedules using that selection strategy.

- **Deterministic Best Response (d_br):** Our presented approach, but limited to a single schedule, rather than over possibly many schedules.

Baseline Comparison

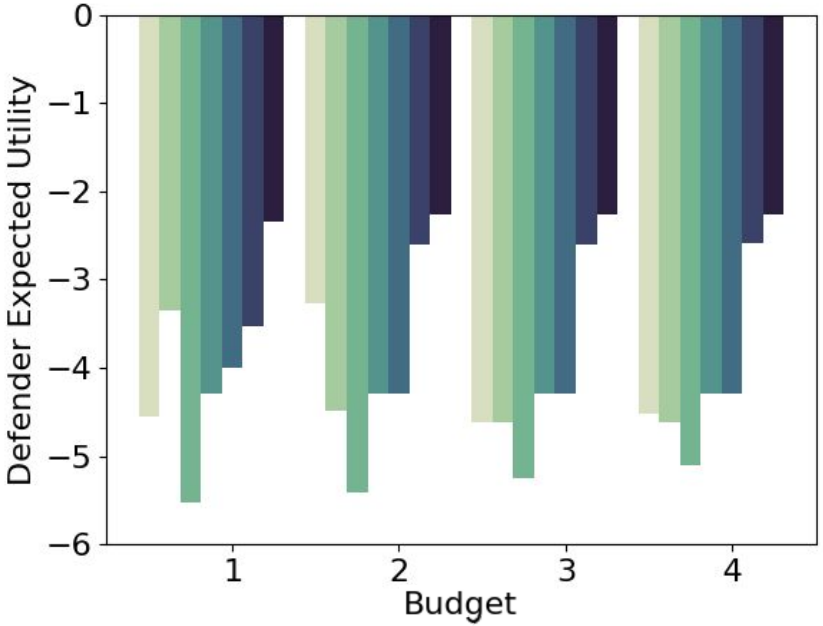
Optimal Randomization



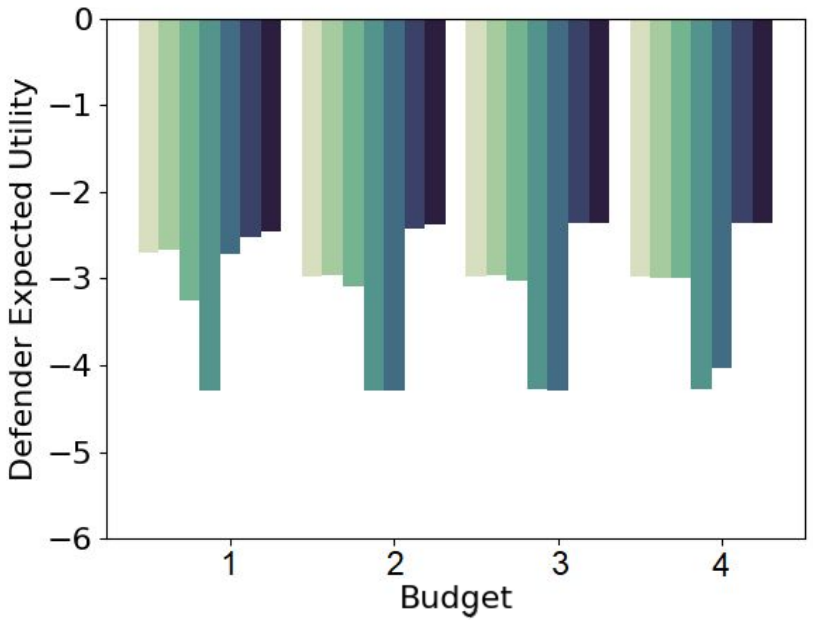
Is randomization always strictly beneficial?

Is randomization strictly beneficial?

All Files



PDFs



How sensitive is our model to changes to the balance between rewards and costs?

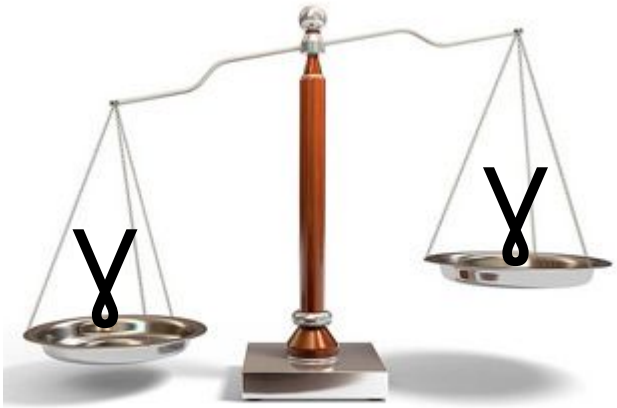
Cost-Weighting Values

Our model explicitly makes a tradeoff between cost and reward for each player.

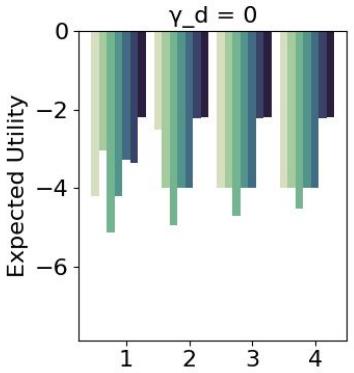
Our model uses variable γ_a and γ_d to represent the trade-off between cost and reward.

Unlike other parameters, these cannot be learned from data easily.

We tested different values for γ_a and γ_d to see if changing them altered the effectiveness of our proposed methods.

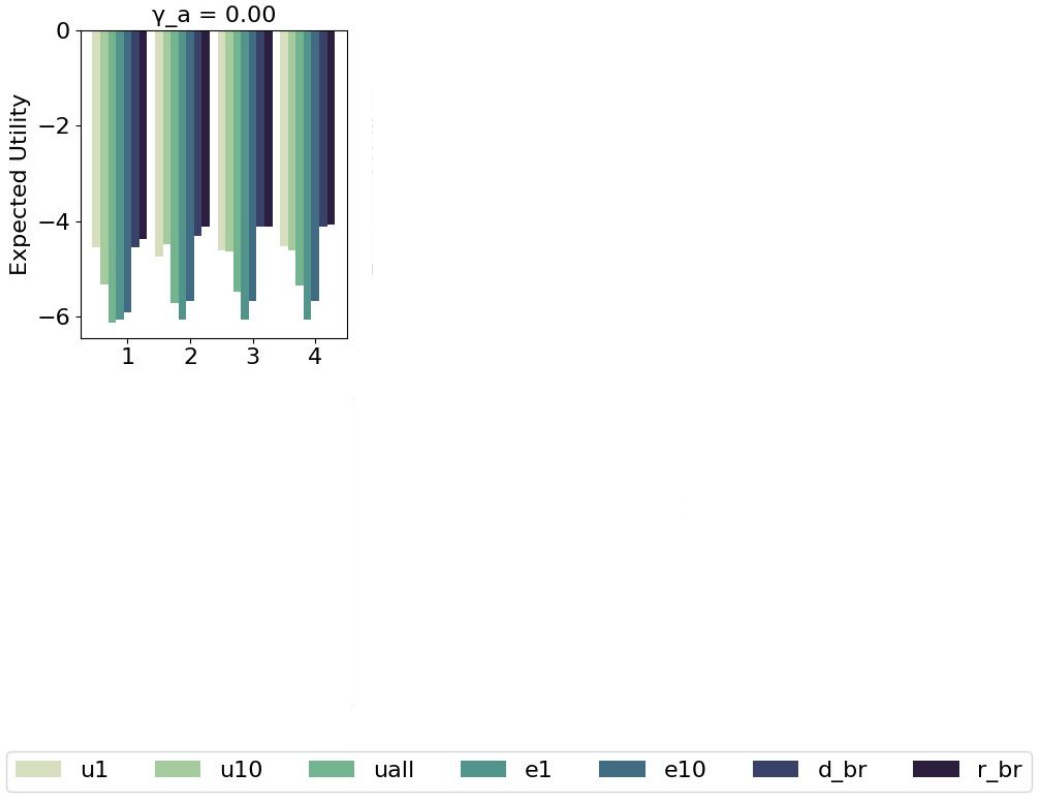


Effects of Changing Defender Cost-Reward Tradeoff

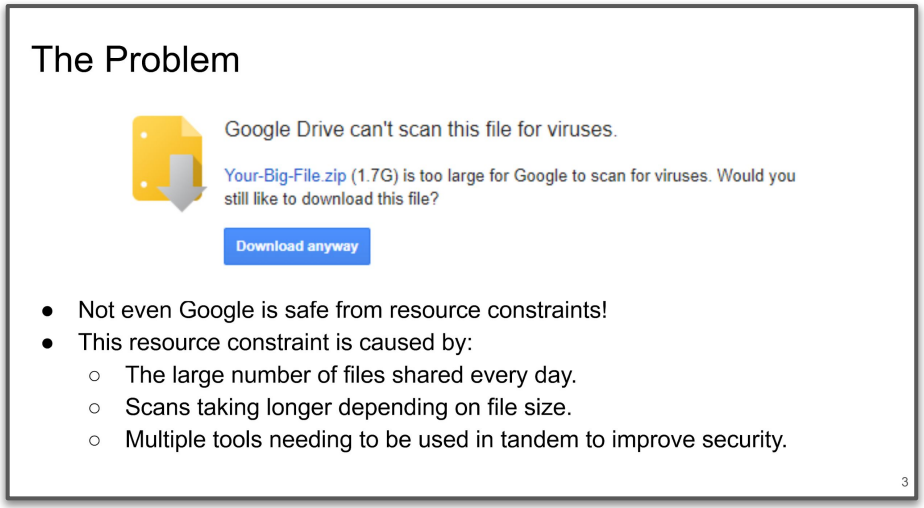


u1 u10 uall e1 e10 d_br r_br

Effects of Changing Attacker Cost-Reward Tradeoff



Conclusion



- Ensuring the safety and security of exchanged files is a growing challenge faced by large software platforms.
  - Platforms are faced with resource constraints.
  - This prevents them from running all available tools.

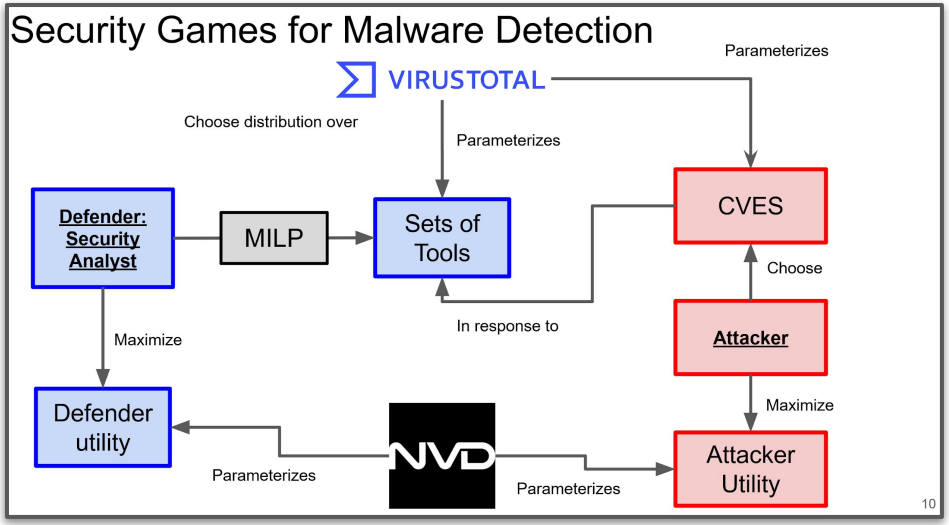
Conclusion

The Problem: How to lower costs and avoid creating weaknesses in our system?

- Scan all files using fewer tools?
  - Attacks are more likely to slip through
- Pick a few strong overall tools?
  - Attacker can learn what tools and analyses you're using
- Pick only the best tools based on which attacks are likely to be made.
  - An attacker could change their actions to exploit this prediction.
- So what should we do?

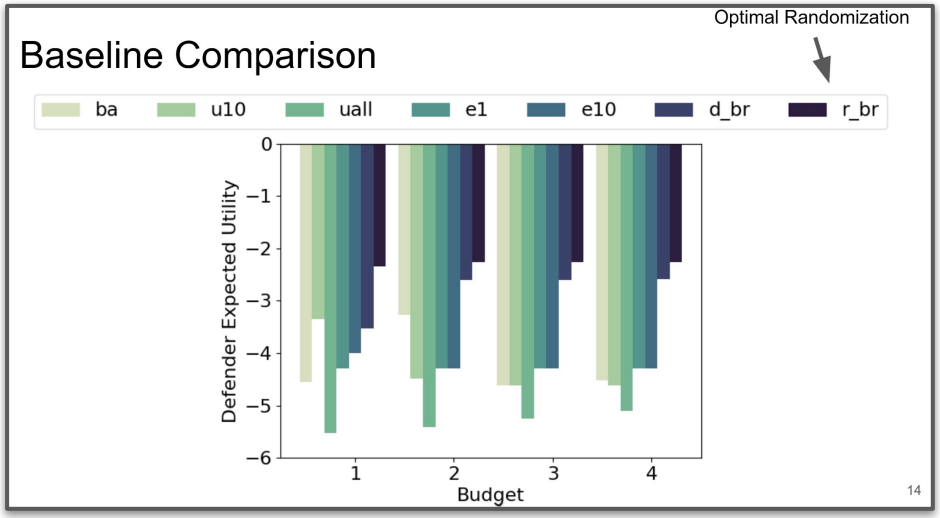
- Choosing how to best run analyses is a complicated task.
  - Different tools use different scanning strategies with different results.
  - Overlap in capabilities of tools can lead to inefficient use of resources.
  - Randomization leads to a reduction in the chance of attackers exploiting gaps in coverage.

Conclusion



- To address these problems we presented an approach that uses stackelberg games.
  - These strategies can be solved for using a MILP.
  - This model was parameterized with real-world data from VirusTotal and NVD.

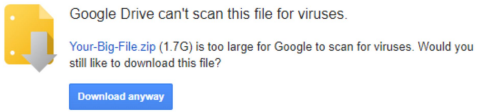
Conclusion



- We evaluated our proposed method against a set of alternate strategies.
  - We found this it does as well as or outperforms all alternate strategies we considered.
  - We found that changing the free parameters in our model do not lead to qualitative differences in our results.

Game Theoretic Malware Detection

The Problem



- Not even Google is safe from resource constraints!
- This resource constraint is caused by:
  - The large number of files shared every day.
  - Scans taking longer depending on file size.
  - Multiple tools needing to be used in tandem to improve security.

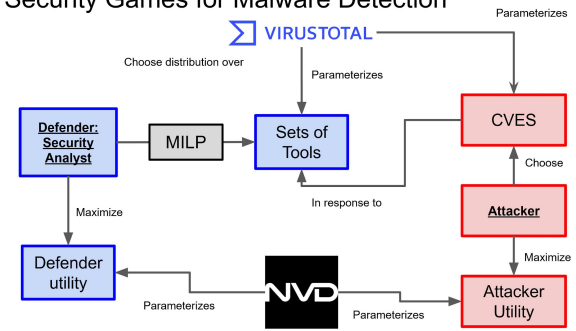
3

The Problem: How to lower costs and avoid creating weaknesses in our system?

- Scan all files using fewer tools?
  - Attacks are more likely to slip through
- Pick a few strong overall tools?
  - Attacker can learn what tools and analyses you're using
- Pick only the best tools based on which attacks are likely to be made.
  - An attacker could change their actions to exploit this prediction.
- So what should we do?

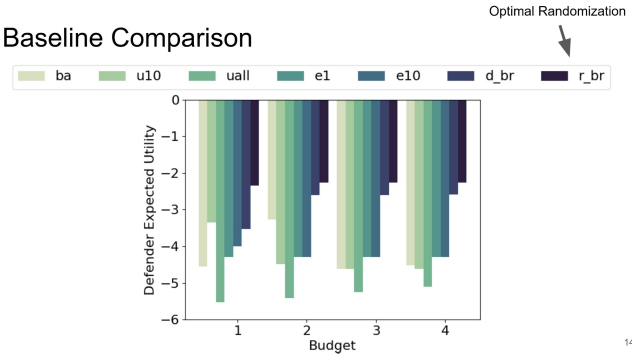
6

Security Games for Malware Detection



10

Baseline Comparison



14

Contact us at: nbombard@ualberta.ca and revan@ualberta.ca

Optimal Defender Strategies

Budget	Utility	Schedule	Probability
1	-2.341	Avira	0.185
		ClamAV	0.024
		ESET NOD32	0.029
		F-Secure	0.019
		TrendMicro	0.119
		ZoneAlarm	0.625
2	-2.270	ClamAV, F-Prot	0.237
		DrWeb, Kaspersky	0.322
		Kaspersky, TACHYON	0.119
		TheHacker, ZoneAlarm	0.322
3	-2.269	Avast-Mobile, ClamAV, F-Prot	0.237
		DrWeb, Kaspersky, TotalDefense	0.441
		DrWeb, Kaspersky, TheHacker	0.203
		Kaspersky, SUPERAntiSpyware, TheHacker	0.119
4	-2.269	Avast-Mobile, ClamAV, F-Prot	0.237
		DrWeb, Kaspersky, TheHacker, TotalDefense	0.763

Stackelberg Games

1. The leader chooses a strategy $s_L \in S_L$.
2. The follower observes s_L.
3. The follower chooses a strategy $s_F \in S_F$.
4. An action profile (a_L, a_F) is sampled with $a_L \sim S_L$ and $a_F \sim S_F$.
5. Each agent i receives utility $u_i(a_L, a_F)$.

$$\text{maximize} \quad \widetilde{U}_d$$

$$\text{subject to} \quad \widetilde{y}_v \in \{0,1\} \quad \forall v \in V$$

$$\sum_{v \in V} \widetilde{y}_v = 1$$

$$0 \leq \widetilde{x}_s \leq 1 \quad \forall s \in S$$

$$\sum_{s \in S} \widetilde{x}_s = 1$$

$$\widetilde{U}_d - \sum_{s \in S} \widetilde{x}_s u_d(s,v) \leq (1 - \widetilde{y}_v) Z \quad \forall v \in V$$

$$0 \leq \widetilde{U}_a - \sum_{s \in S} \widetilde{x}_s u_a(s,v) \leq (1 - \widetilde{y}_v) Z \quad \forall v \in V$$

- Schedules ~ S
- Detection probabilities ~ $p(s,v)$
- Rewards ~ r_a, r_d
- Vulnerabilities ~ V
- Costs ~ c_a, c_d

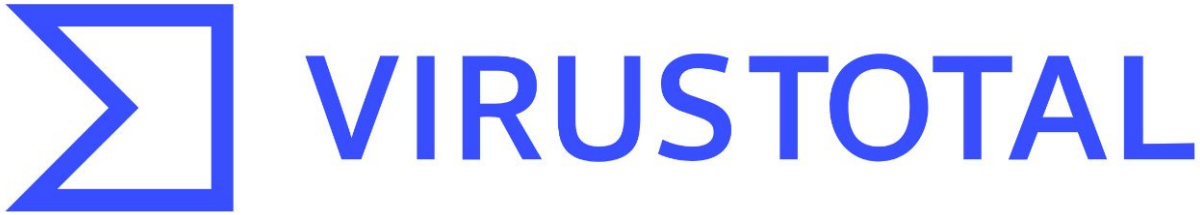
Type	Budget	Load Parameters	Generate MILP	CPLEX Read MILP	CPLEX Presolve	CPLEX Probe	CPLEX Solve
All	1	0.397 ± 0.021	0.055 ± 0.003	0.010 ± 0.000	0.035 ± 0.009	0.006 ± 0.009	0.207 ± 0.110
	2	0.408 ± 0.019	1.759 ± 0.118	0.245 ± 0.016	1.306 ± 0.290	0.093 ± 0.025	3.447 ± 1.349
	3	0.916 ± 0.033	36.934 ± 4.855	5.562 ± 0.921	32.118 ± 8.573	0.217 ± 0.064	66.400 ± 32.553
	4	14.313 ± 0.691	523.592 ± 91.450	84.752 ± 20.274	457.395 ± 135.952	1.249 ± 0.384	873.180 ± 391.370
PDF	1	0.385 ± 0.010	0.018 ± 0.001	0.005 ± 0.005	0.012 ± 0.004	0.000 ± 0.000	0.114 ± 0.042
	2	0.389 ± 0.004	0.482 ± 0.058	0.069 ± 0.010	0.283 ± 0.071	0.035 ± 0.025	0.975 ± 0.552
	3	0.593 ± 0.008	8.188 ± 1.700	1.150 ± 0.272	5.963 ± 1.859	0.081 ± 0.039	14.171 ± 5.515
	4	6.429 ± 0.607	85.299 ± 18.998	13.678 ± 4.018	68.415 ± 23.252	0.285 ± 0.109	155.310 ± 59.682

Table 3: The runtime of all the stages in our pipeline (in seconds). Load parameters is the time needed to load model parameters from VirusTotal and generate utility matrices for the defender and attacker. Generate MILP is the time needed to write the MILP to a file. Read MILP is the time to read the MILP into CPLEX. CPLEX Presolve and Probe are CPLEX optimization stages that reduce the size of the MILP before solving. Solve is the wall time to solve the MILP.

Baseline Strategy	Budget	Solve Time (s)
ba	1	0.00024 ± 0.00003
	2	0.00121 ± 0.00031
	3	0.03064 ± 0.00588
	4	0.69511 ± 0.09305
u10	1	0.0003 ± 0.00002
	2	0.00093 ± 0.00002
	3	0.02670 ± 0.00034
	4	0.65883 ± 0.07263
uall	1	0.00017 ± 0.00002
	2	0.00116 ± 0.00004
	3	0.02723 ± 0.00185
	4	0.62969 ± 0.13856
e1	1	0.00066 ± 0.00023
	2	0.00225 ± 0.00055
	3	0.04842 ± 0.00077
	4	1.12192 ± 0.13744
e10	1	0.00037 ± 0.00002
	2	0.00165 ± 0.00007
	3	0.04780 ± 0.00122
	4	1.12371 ± 0.12602
d_br	1	0.00012 ± 0.00002
	2	0.00120 ± 0.00048
	3	0.03026 ± 0.00408
	4	0.71663 ± 0.12118

Table 4: Runtime to solve games for baseline strategies, averaged over 12 runs. Baselines share parameter loading, which has a runtime of 0.3967 ± 0.0205 for budget 1, 0.4079 ± 0.0191 for budget 2, 0.9158 ± 0.0334 for budget 3 and 14.3128 ± 0.6912 for budget 4. We detail the baseline strategy abbreviations in Section 5.1.

Data Resources: VirusTotal

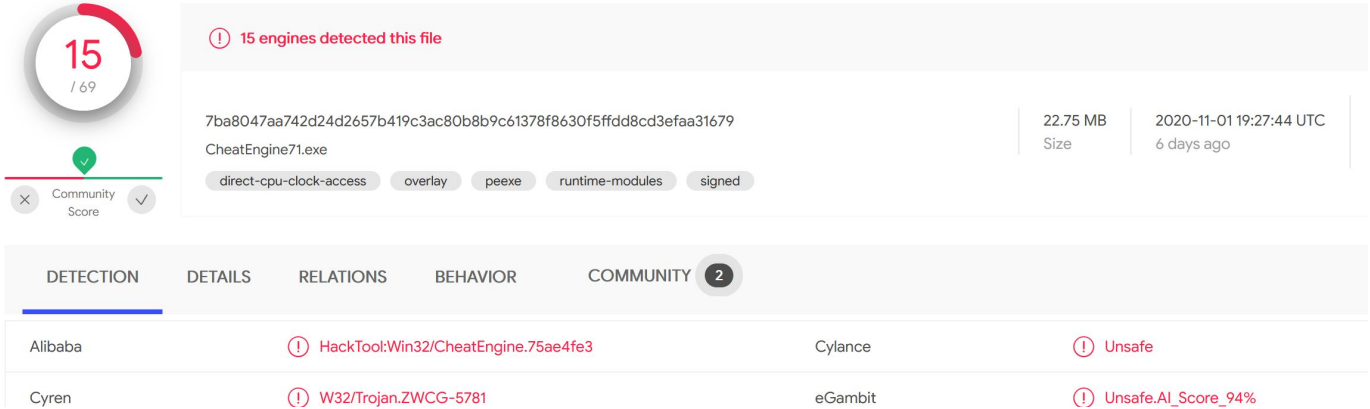


VirusTotal is an online platform that allows users to test any file for malware.

They also offer researchers access to a dataset of over 30,000 malware files and the results of malware scans of over 86 malware detection tools on those files.

This dataset allows us to understand how antimalware tools perform against real malicious software.

What does VirusTotal give us?



Among the information contained in each of the 30,000+ malware scan results files was:

- For each tool, whether the file was flagged as malicious.
- The Common Vulnerabilities and Exposures (CVE) tag, denoting the type of malware that the file contains.

National Vulnerability Database (NVD)

We can use the CVE tags to look up attacks to the National Vulnerability Database, or NVD.

NVD provides a database indexed by CVE tag, which provides numeric representations of:

- The exploitability of vulnerabilities
  - We use this to represent the cost to attackers c_a associated with attempting to exploit a vulnerability.
- The harm each vulnerability can cause if exploited.
  - We use this to represent the r_a, r_d rewards for when a vulnerability is exploited.

Cost to defender: False Positives

The tools in VirusTotal flagged between 25%-70% of all obfuscated, non-malicious files as malicious, according to a study by Zhu et al [1].

Having a tool falsely flag a non-malicious file imposes the risk of annoying end-users, so we represent the cost of using each tool with the false positive rate of each tool.

We calculate these rates by measuring how often each tool falsely flagged a safe file within the dataset provided alongside Zhu et al.’s publication. We then generalize these results to schedules.

[1] https://www.usenix.org/conference/usenixsecurity20/presentation/zhu