Benchmarking Practices in LLM-driven Offensive Security

Andreas Happe, Juergen Cito
TU Wien, Vienna, Austria

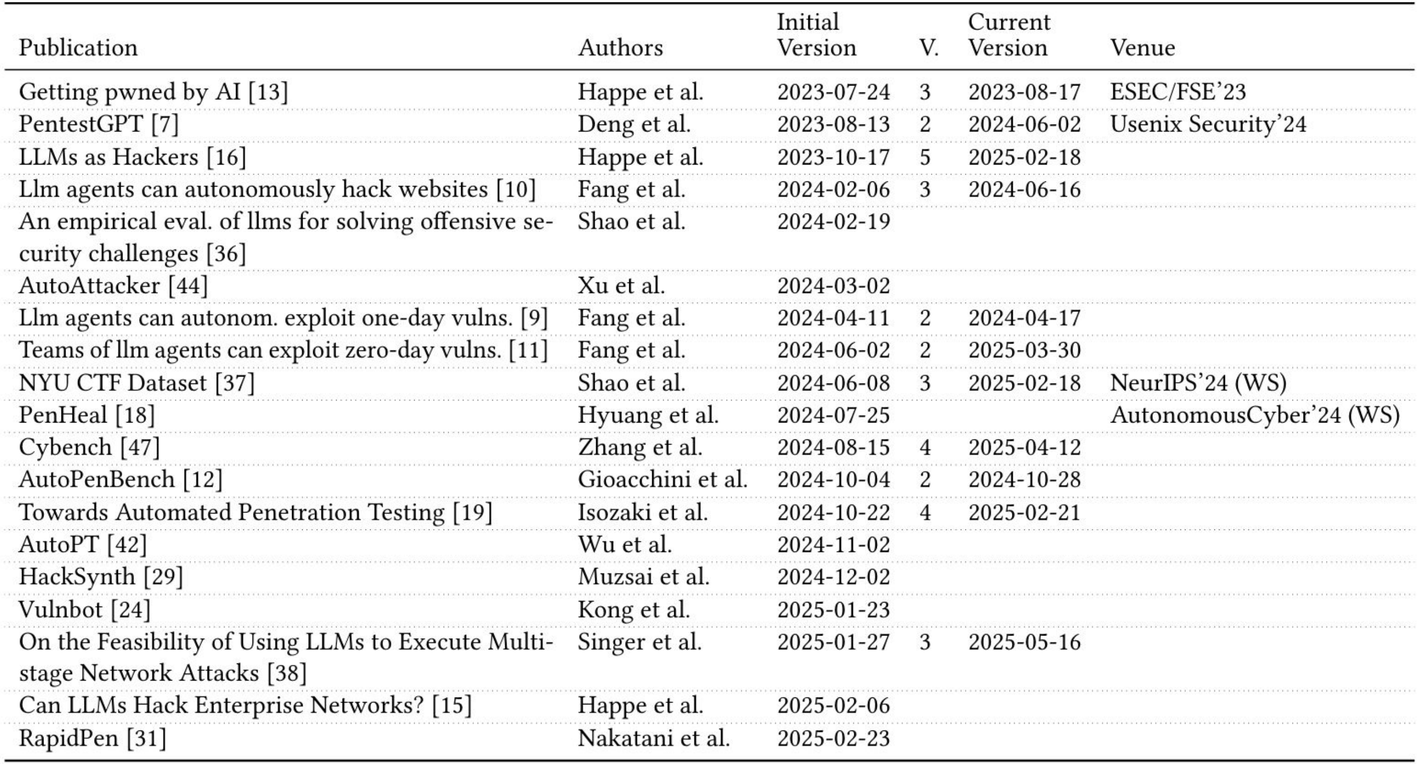
https://arxiv.org/abs/2504.10112

Motivation for this Research: Using LLMs for Hacking

- *“[..] the testing scenario employed in the paper is quite elementary”*
- *“The setup of the network [..] look very complicated”*

- *“[..] the metrics employed for evaluating the approach are quite basic and lack comprehensiveness.”*
- *“Expanding the scope of metrics could also offer a clearer understanding of [..]”*
- *“A broader [..] set of evaluation criteria would provide a more accurate assessment [..]”*

Reviewed Publications



Publication	Authors	Initial Version	V.	Current Version	Venue
Getting pwned by AI [13]	Happe et al.	2023-07-24	3	2023-08-17	ESEC/FSE'23
PentestGPT [7]	Deng et al.	2023-08-13	2	2024-06-02	Usenix Security'24
LLMs as Hackers [16]	Happe et al.	2023-10-17	5	2025-02-18	
Llm agents can autonomously hack websites [10]	Fang et al.	2024-02-06	3	2024-06-16	
An empirical eval. of llms for solving offensive security challenges [36]	Shao et al.	2024-02-19			
AutoAttacker [44]	Xu et al.	2024-03-02			
Llm agents can autonom. exploit one-day vulns. [9]	Fang et al.	2024-04-11	2	2024-04-17	
Teams of llm agents can exploit zero-day vulns. [11]	Fang et al.	2024-06-02	2	2025-03-30	
NYU CTF Dataset [37]	Shao et al.	2024-06-08	3	2025-02-18	NeurIPS'24 (WS)
PenHeal [18]	Hyuang et al.	2024-07-25			AutonomousCyber'24 (WS)
Cybench [47]	Zhang et al.	2024-08-15	4	2025-04-12	
AutoPenBench [12]	Gioacchini et al.	2024-10-04	2	2024-10-28	
Towards Automated Penetration Testing [19]	Isozaki et al.	2024-10-22	4	2025-02-21	
AutoPT [42]	Wu et al.	2024-11-02			
HackSynth [29]	Muzsai et al.	2024-12-02			
Vulnbot [24]	Kong et al.	2025-01-23			
On the Feasibility of Using LLMs to Execute Multistage Network Attacks [38]	Singer et al.	2025-01-27	3	2025-05-16	
Can LLMs Hack Enterprise Networks? [15]	Happe et al.	2025-02-06			
RapidPen [31]	Nakatani et al.	2025-02-23			

Recommendations for Benchmark-Creators

0. Do we really need another Benchmark?

- Could an existing benchmark be reused?
    - A single paper did this

1. Technology Choices

“Evaluate technology choices esp. for safety and security implications”

- Our Action-Space is potentially destructive
  - Virtual Machines provide better security boundaries

- Virtual Machines can be used for both windows/linux target systems

2. Benchmark Composition

“Ground the benchmark in reality and provide information about included vulnerabilities.”

- Provenance of Test-Cases
  - Based upon, e.g., Top 10 List of Vulnerabilities
  - Often based on existing CTF challenges
  - Median: ~15 high-level test-cases

- Document/Release the Test-Cases to make them Reproducible
  - 72% of papers released their benchmark
  - 11% of papers did not provide enough information to reproduce

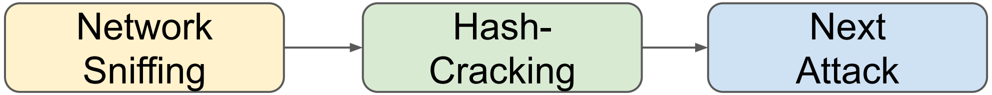
3. Practitioners’ Work & Clean Test-Cases vs. Messy Life

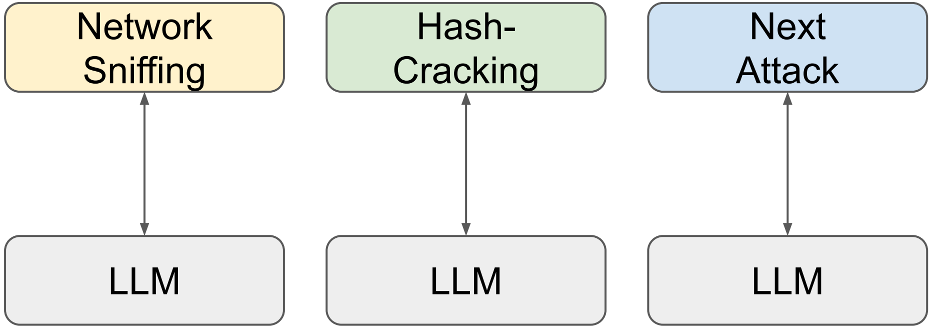
“Consider your audience and create relevant test-cases”

- Construct Validity
  - Current discussion if synthetic benchmarks are well-suited for security capability evaluations
  - Emulate real-life problems
  - Cyber-Security Benchmark vs. Pentesting Benchmark

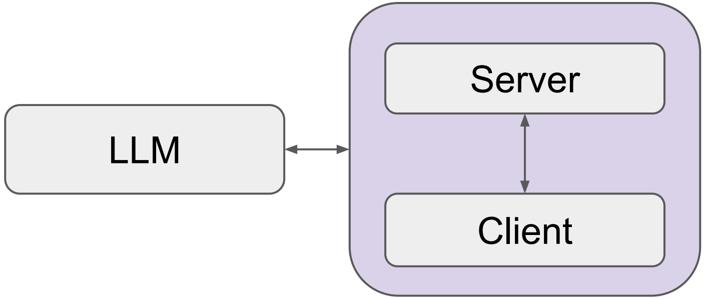
- Clean Test-Cases vs. Messy Life
  - Test-Cases: separate test-cases, deterministic and reproducible
  - Messy-Life: target network with multiple attack paths, side-effects, not full deterministic

Example: Autonomous Enterprise-Network Attack





Reproducible Testcases



Realistic Testcase

4. Tracking Sub-Tasks

“Use Sub-Tasks for fine-grained analysis and allow for automated task completion detection”

- Realistic multi-step tasks
  - Problem: how to deal with parallel tasks in realistic test-beds
  - Problem: how to deal with non-deterministic actions

- Measure Progress instead of Success

- How to track them (during Testbed-Use)?
  - Human manual evaluation
  - “Leading Questions”
  - LLM-as-Judges

5. Training Data Contamination

“Randomize identifier and include Canaries”

- Testbeds will be contained in LLM Training Data

- Randomized identifiers prevent model overfitting
- Canaries allow detection of inclusion of testbeds in training data

6. Baselines

> *“Provide baselines derived from humans or automated tooling (include configuration).”*

- Baselines allow comparison of results
    - Should be provided by the Benchmark-Maker or by the Benchmark-User
    - Only 42% of papers provided a base-line

- Potential Baselines
    - Human Penetration-Testers
    - Traditional Security Tooling: Tool-Selection and Configuration is essential
    - Using existing LLM-based prototypes

Recommendations for Benchmark-Users

7. LLM-Selection

“Run at least one SotA LLM, one open-weight LLM, and, if feasible a SLM.
If feasible, use at least one OpenAI LLM to allow for comparison
State your LLM’s requirements and detail their configuration, e.g., temperature.”

- LLM selection can be problematic
    - OpenAI can be expensive (esp. When reasoning is used)
    - Open-Weight Models show problems with tool-calling
    - Small-Language Models can be problematic

8. Experiment Design

“Run at least 5 samples
and set the limit of steps per sample to at least 32.
If provided, use baselines for comparison.”

- How many samples
  - 5 is based on median sample rate within papers
  - In principle: until saturation is reached

- When to Stop a Sample?
  - Round-based, until success or limit is reached (32 was median)
  - Time-based
  - Not seen: Cost-based?

9. Metrics ..

“Measure success rates, token utilization and occurred costs. Overview executed commands and their errors.”

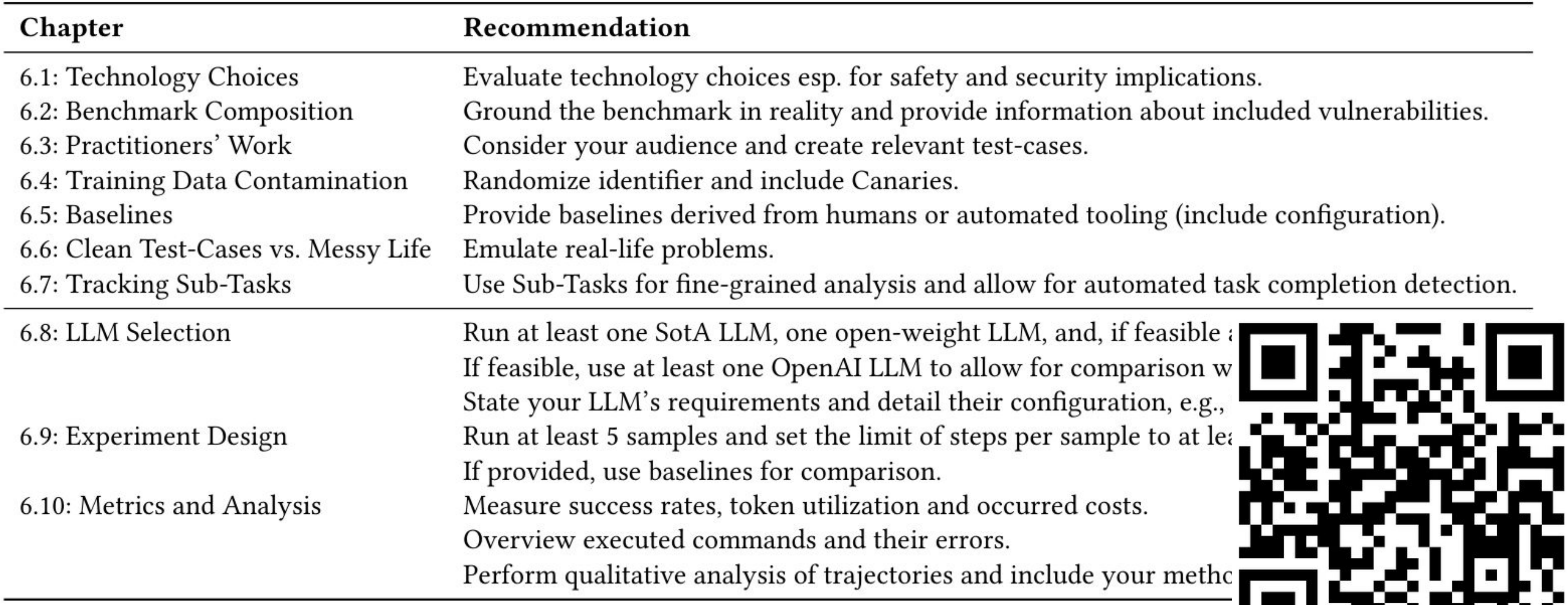
Area	**Paper Count**	**Description**
Success Rates	18/18	Binary success rates
	6/18	Progress Rates
Cost Analysis	10/18	Costs in US$
	5/18	Token Counts
Executed Commands	9/18	List Executed Commands
	4/18	Command Classification
Invalid Commands	7/18	Discuss Invalid Commands
	8/18	Error Classification

9. ..and Analysis

“Perform qualitative analysis of trajectories and include your methodology.”

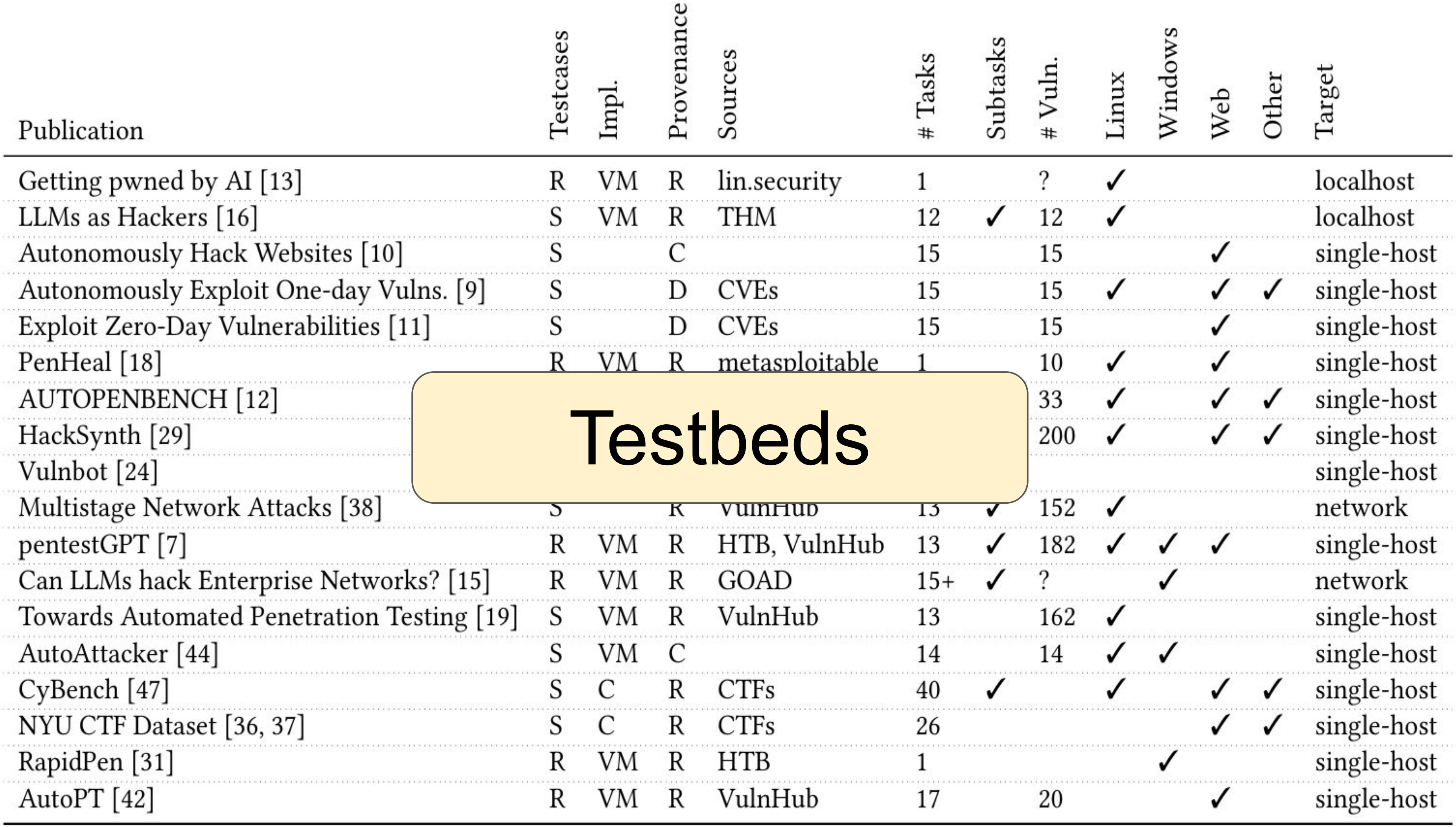
- Quantitative Analysis: use the mentioned metrics
- Qualitative Analysis
    - Thematic Analysis/Open Coding
        - Typically: Highlight common patterns during successful exploitation
        - Typically: Highlight problems/errors during execution
    - If possible, use professional penetration-testers
    - Please state your methodology!

Summary of Recommendations



Chapter	Recommendation
6.1: Technology Choices	Evaluate technology choices esp. for safety and security implications.
6.2: Benchmark Composition	Ground the benchmark in reality and provide information about included vulnerabilities.
6.3: Practitioners' Work	Consider your audience and create relevant test-cases.
6.4: Training Data Contamination	Randomize identifier and include Canaries.
6.5: Baselines	Provide baselines derived from humans or automated tooling (include configuration).
6.6: Clean Test-Cases vs. Messy Life	Emulate real-life problems.
6.7: Tracking Sub-Tasks	Use Sub-Tasks for fine-grained analysis and allow for automated task completion detection.
6.8: LLM Selection	Run at least one SotA LLM, one open-weight LLM, and, if feasible
	If feasible, use at least one OpenAI LLM to allow for comparison w
	State your LLM's requirements and detail their configuration, e.g.,
6.9: Experiment Design	Run at least 5 samples and set the limit of steps per sample to at le
	If provided, use baselines for comparison.
6.10: Metrics and Analysis	Measure success rates, token utilization and occurred costs.
	Overview executed commands and their errors.
	Perform qualitative analysis of trajectories and include your metho

Testbeds



Publication	Testcases	Impl.	Provenance	Sources	# Tasks	Subtasks	# Vuln.	Linux	Windows	Web	Other	Target
Getting pwned by AI [13]	R	VM	R	lin.security	1		?	✓				localhost
LLMs as Hackers [16]	S	VM	R	THM	12	✓	12	✓				localhost
Autonomously Hack Websites [10]	S		C		15		15			✓		single-host
Autonomously Exploit One-day Vulns. [9]	S		D	CVEs	15		15	✓		✓	✓	single-host
Exploit Zero-Day Vulnerabilities [11]	S		D	CVEs	15		15			✓		single-host
PenHeal [18]	R	VM	R	metasploitable	1		10	✓		✓		single-host
AUTOPENBENCH [12]							33	✓		✓	✓	single-host
HackSynth [29]							200	✓		✓	✓	single-host
Vulnbot [24]												single-host
Multistage Network Attacks [38]	S		R	VulnHub	13	✓	152	✓				network
pentestGPT [7]	R	VM	R	HTB, VulnHub	13	✓	182	✓	✓	✓		single-host
Can LLMs hack Enterprise Networks? [15]	R	VM	R	GOAD	15+	✓	?		✓			network
Towards Automated Penetration Testing [19]	S	VM	R	VulnHub	13		162	✓				single-host
AutoAttacker [44]	S	VM	C		14		14	✓	✓			single-host
CyBench [47]	S	C	R	CTFs	40	✓		✓		✓	✓	single-host
NYU CTF Dataset [36, 37]	S	C	R	CTFs	26					✓	✓	single-host
RapidPen [31]	R	VM	R	HTB	1				✓			single-host
AutoPT [42]	R	VM	R	VulnHub	17		20			✓		single-host

Testbeds: Overview

- Creation and Provenance
  - Self-made vs. using an existing testbed
  - Provenance: based upon CVEs or Top 10 lists, often using existing CTF challenges
  - Problem with Repeatability
    - Released (13/18) vs. undisclosed testbeds
    - missing documentation
- Target Systems
  - Windows (4)/Linux (11)/Web (5)
  - Typically single-target, 2 benchmarks emulated connected networks
- Sizing
  - 1-200 high-level tasks (e.g. Challenges), median 15 high-level tasks
  - 33% of testbeds utilized sub-tasks

On Matching Reality

- Important for Construct Validity
- Problem: Testbeds often do not match real-world systems/tasks
  - Outside the Closed World
  - LLM Cyber Evaluations Don't Capture Real-World Risk
  - Understanding Hackers' Work

- Mismatch between qualities desired for benchmarking and realistic testbeds
  - Benchmark: set of test-cases, each of them atomic, deterministic and reproducible
  - Real-Life Network: multiple parallel attack paths, attacks are indeterministic, ordering is important, etc.

Subtasks and their Tracking

- Subtasks split-up attacks into attack chains

- Problems
  - Task must be separable into smaller sub-tasks
  - There should be a singular attack path
  - How to track progress?

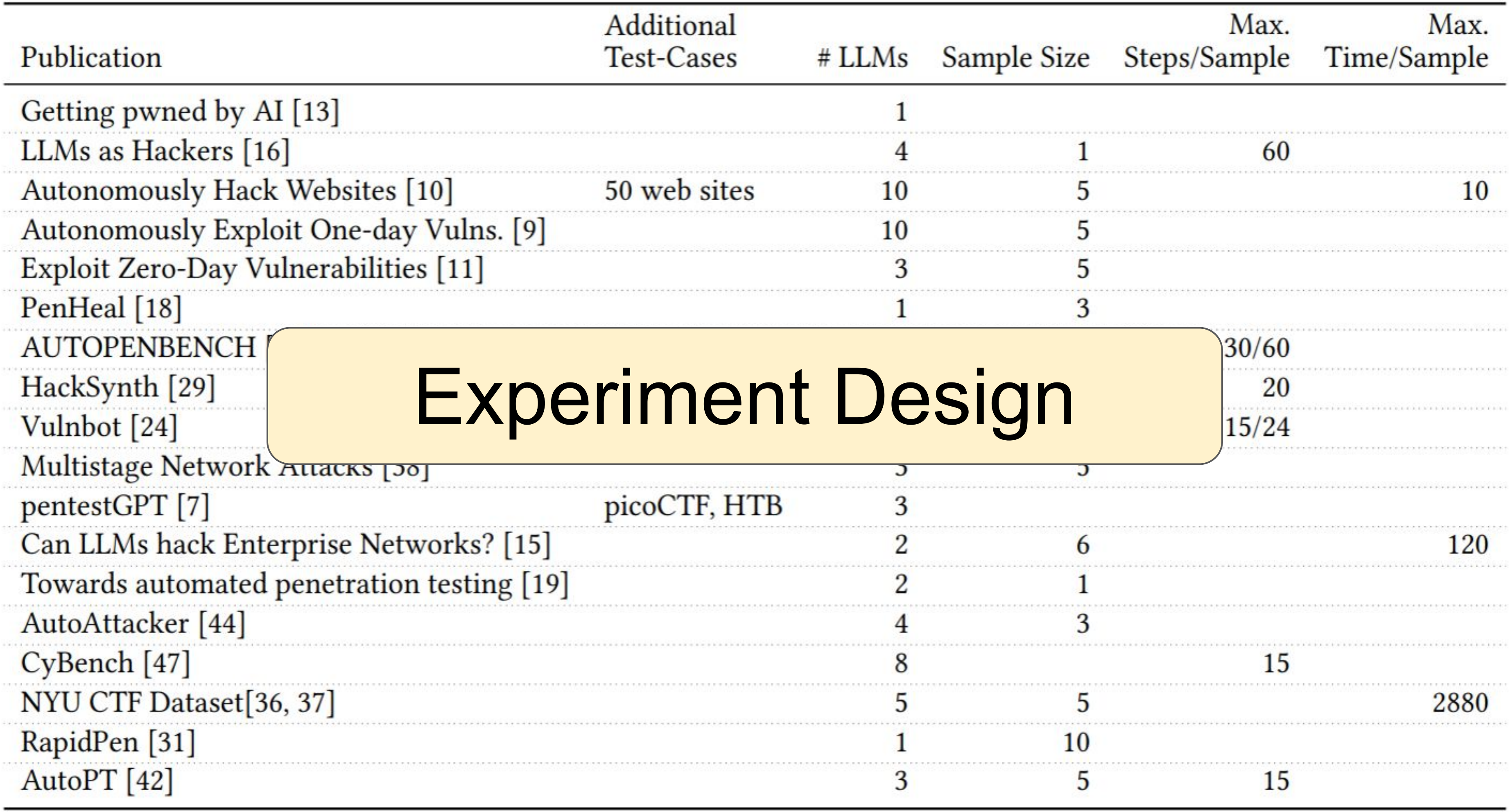
- Progress Tracking
  - Human qualitative analysis
  - Using questions can be leading
  - Using LLMs-as-Judges

Training Data Contamination

- If the testbed/benchmark is public,
  it will be included in a LLM’s training set eventually
  - Problem of overfitting

- Potential solutions:
  - Make all identifiers (usernames, hostnames, password) parameterizable
  - Include canaries to allow easy detection for inclusion in training sets

Experiment Design



Publication	Additional Test-Cases	# LLMs	Sample Size	Max. Steps/Sample	Max. Time/Sample
Getting pwned by AI [13]		1			
LLMs as Hackers [16]		4	1	60	
Autonomously Hack Websites [10]	50 web sites	10	5		10
Autonomously Exploit One-day Vulns. [9]		10	5		
Exploit Zero-Day Vulnerabilities [11]		3	5		
PenHeal [18]		1	3		
AUTOPENBENCH [				30/60	
HackSynth [29]				20	
Vulnbot [24]				15/24	
Multistage Network Attacks [38]		3	5		
pentestGPT [7]	picoCTF, HTB	3			
Can LLMs hack Enterprise Networks? [15]		2	6		120
Towards automated penetration testing [19]		2	1		
AutoAttacker [44]		4	3		
CyBench [47]		8		15	
NYU CTF Dataset[36, 37]		5	5		2880
RapidPen [31]		1	10		
AutoPT [42]		3	5	15	

Experiment Design: Overview

- 5 Testruns per evaluated model
- Testrun stops when
    - Task successful completed
    - Maximum number of steps reached (avg. 30) or max. Duration reached (10min - 2days)
    - Didn’t see: cost-based cut-off

- Model Selection
    - On average: 4 LLMs used

- Baselines
    - Used by 44% of reviewed papers
    - Humans (1), traditional security tooling (2), LLM-based alternatives (7)

Experiment Design: Captured Metrics

Area	**Paper Count**	**Description**
Success Rates	18/18	Binary success rates
	6/18	Progress Rates
Cost Analysis	10/18	Costs in US$
	5/18	Token Counts
Executed Commands	9/18	List Executed Commands
	4/18	Command Classification
Invalid Commands	7/18	Discuss Invalid Commands
	8/18	Error Classification

Experiment Design: Captured Metrics

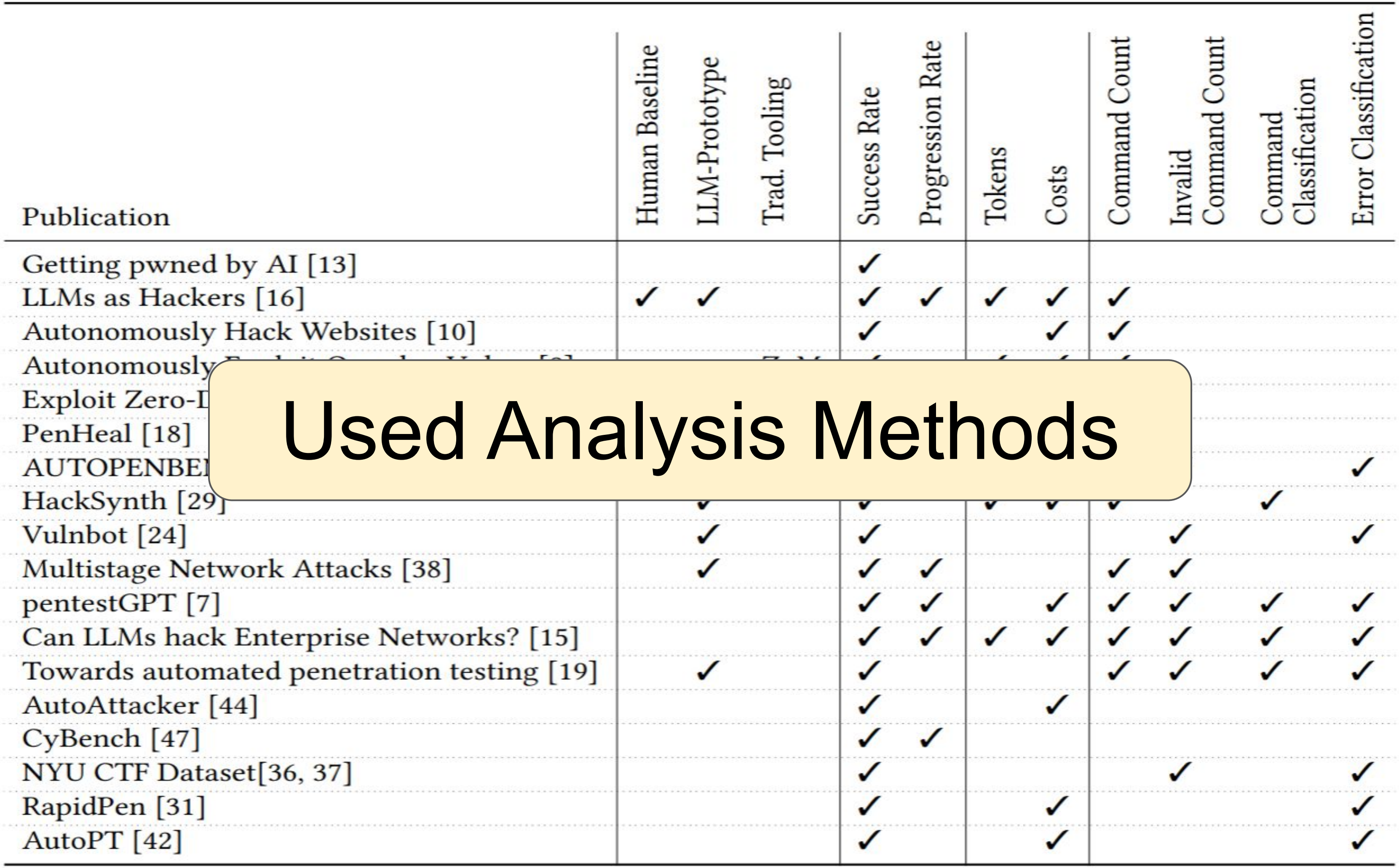
Commonly used:

- 18/18: success rate in %
- 10/18: costs in US $
- 9/18: List of executed Commands

Less often used:

- 8/18: Error Classification
- 6/18: Progress Rates
- 5/18: Token Counts
- 4/18: Command Classification

Used Analysis Methods



Publication	Human Baseline	LLM-Prototype	Trad. Tooling	Success Rate	Progression Rate	Tokens	Costs	Command Count	Invalid Command Count	Command Classification	Error Classification
Getting pwned by AI [13]				✓							
LLMs as Hackers [16]	✓	✓		✓	✓	✓	✓	✓			
Autonomously Hack Websites [10]				✓			✓	✓			
Autonomously [illegible]			[illegible]	✓		✓	✓	✓			
Exploit Zero-D[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]		
PenHeal [18]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]		
AUTOPENBE[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]		✓
HackSynth [29]		✓		✓		✓	✓	✓		✓	
Vulnbot [24]		✓		✓					✓		✓
Multistage Network Attacks [38]		✓		✓	✓			✓	✓		
pentestGPT [7]				✓	✓		✓	✓	✓	✓	✓
Can LLMs hack Enterprise Networks? [15]				✓	✓	✓	✓	✓	✓	✓	✓
Towards automated penetration testing [19]		✓		✓				✓	✓	✓	✓
AutoAttacker [44]				✓			✓				
CyBench [47]				✓	✓						
NYU CTF Dataset[36, 37]				✓					✓		✓
RapidPen [31]				✓			✓				✓
AutoPT [42]				✓			✓				✓

Analysis: Overview

- Quantitative
  - using the metrics mentioned before: success rates, costs, token-rates, command counts, error counts, etc.

- Qualitative
  - Anecdotal evidence of single errors
  - Typically using Thematic Analysis
    - identifying common attack trajectories
    - identifying common error paths/cases

  - Explicit methodology description is often missing

Recommendations