International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 10 | Oct - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

"A Gambit Against the Kill Chain: When Game Theory Meets the Adversary's Next Move"

Tanmay S. Gawankar,
Cyber Security Department,
Patkar Varde College.

A silent revolution is happening, as one of the oldest games of kings gets into the digital battlefield of modern cybersecurity. We are at the dawn of a new era of cyber defense, in which traditional protection systems only respond to attacks, artificial intelligence can now predict entire chains of attacks. The cybersecurity game has entered a whole new level.

ABSTRACT:

In the modern age of technology, conventional computer security systems are like an amateur chess player learning to play chess. They are only able to react to security breaches after an intruder has already invaded the network. This creates vulnerabilities for companies whose networks are under siege by sophisticated and well planned cyberattacks. This paper presents an alternative approach to network security that uses the analogy of computer networks as a game of chess. This approach assigns values to different nodes in a network based on their importance within the network, much like different chess pieces having different values on the board. The central domain controllers that coordinate the network are analogous to the king, queens, and rooks in chess and are given higher values than the employee computers, which are the pawns of the network. By using an established chess-playing artificial intelligence algorithm called Minimax with Alpha-Beta pruning, this system is able to mimic a human grandmaster being several moves ahead of an opponent during a game of chess.

The computer system mimics this ability and is designed to determine where a hacker is most likely to strike next before they even arrive at a specific node in the network. A working prototype of the pipeline — from adversary emulation through Sysmon telemetry capture, ground-truth ATT&CK technique labeling, and a search-based evaluation engine — was implemented and validated end-to-end on real, non-synthetic attack telemetry generated in a controlled testbed. This initial evaluation demonstrates the feasibility of the proposed architecture: the pipeline runs end-to-end on real, non-synthetic attack telemetry, and the resulting technique sequences support a first, paired test of the central hypothesis. However, that test does not support the hypothesis: on 35 next-step predictions drawn from two sequences, the depth-3 minimax engine and a static kill-chain baseline performed identically, and a paired significance test found no difference between them. This paper therefore does not demonstrate that game-theoretic lookahead outperforms reactive, rule-based detection; it reports a working testbed and evaluation protocol, an honest first test of the idea, and the specific design and data limitations — discussed in full — that a larger, statistically powered study would need to address before that broader claim could be made.

I. INTRODUCTION:

The digital world has become a battleground in which traditional defenses have proven ineffective against well-prepared hackers. Traditional Security information and event management (SIEM) takes real-time action in the network in a fundamentally reactive way. This is similar to how an amateur chess player would move alone when one of the computers on his network raises a flag. In contrast, hackers are likely to combine multiple approaches and probe the network for weaknesses, bypassing individual defenses and moving from one user endpoint to another. The defensive response is necessarily reactive but is too limited to keep up with the rapid evolution of cyber conflicts in the modern age. Cybersecurity analysts must think not like an amateur chess player but rather like a Chess Grandmaster to get ahead of hackers.

Chess has long been regarded as the ultimate game of strategy, in which the best players are not those who can best react to their opponent's moves but those who can most skill-fully plan their moves several steps ahead. If we consider computer networks as chessboards on which combatants compete for dominance, defenders must think strategically about how to protect their assets. The most successful approach would be to think of network infrastructure assets as chess pieces with similar strategic roles and tactical values. Thus, domain controllers are kings, databases are queens, rooks, and other critical network assets are bishops, while endpoints are pawns.

The purpose of this study is to describe how cybersecurity analysts might use Minimax search with Alpha-Beta Pruning to get ahead of hackers. The research will demonstrate that by using the taxonomies of ATT&CK network attacks as game states, it will be possible to create a cybersecurity engine that would analyze the network attack path three moves ahead, searching for the best possible options for attackers, as a first step toward plugging security gaps before real attackers reach them. Thus, cybersecurity analysts will be able to think several moves ahead like chess grandmasters and chess kings.

The central hypothesis is as follows: H_1: a game-theoretic minimax lookahead search (depth $d \geq 3$) achieves significantly higher next-step attacker trajectory prediction accuracy (Acc_K) than traditional static or rule-based intrusion detection. The null hypothesis, $H_{0,1}$, is that there is no statistically significant difference between the two ($Acc_{Minimax} = Acc_{Static}$).

We test H_1 on a small pilot. The study makes four contributions. (1) A reproducible testbed in which MITRE Caldera [1] emulates adversary operations against an instrumented Windows endpoint, with a pipeline that converts Sysmon [2] telemetry into ATT&CK-labeled technique sequences. (2) A depth-3 alpha-beta minimax engine, a static kill-chain baseline, and a paired walk-forward protocol for comparing them. (3) A test of H_1 on 35 prediction steps, which does not support the hypothesis, together with an analysis of why the design and the data limit what this pilot can show. (4) A list of data-handling problems we ran into, which are likely to affect similar studies. We do not claim that the individual components are new; the aim is to report an honest first test of their combination.

II. LITERATURE REVIEW:

The development of methods used to protect enterprises has been a perpetual process. It started from simply being able to detect events after they occurred to the point of being able to forecast potential problems. This paper will discuss the current approaches to network vulnerability modeling, game theory, intrusion detection, and chess game methods; and provide context to the research gap that the following work attempts to bridge.

A. *Classical Attack Graphs And Network Vulnerability Modeling:*

In this paper, a move is defined as the use of a single MITRE ATT&CK [3] technique that allows the adversary to change its compromised network node to an adjacent one reachable by it on the graph G. An early attempt at representing multiple-stage cyber-attacks consisted of static attack graphs capturing reachability and software vulnerability information between network assets. Attack graphs produced by existing methodologies, such as MulVAL [4], have a well-known "path explosion" problem, where the number of paths increases rapidly with the size of the network under analysis; subsequent work has focused specifically on scaling attack-graph generation to larger networks [5].

More importantly, static attack graphs summarize the reachability information in a graph structure that does not update as the network environment changes. In other words, these structures represent a freeze of the network state. At the same time, the most informative attack graphs consider the dynamic nature of network activity during an ongoing attack and do not allow for easy updates. For example, in the presence of a moving attacker, the changing state of the network should be constantly updated, which static attack graphs do not allow.

B. *Chess Game Theory in Cybersecurity: A Literature Review*

To address the complex and challenging nature of the cyber domain, researchers and scientists are turning to the concept of game theory. In particular, past research efforts have focused on using Stackelberg security games, MDPs, and zero-sum matrix games for the purposes of developing a relationship between a pursuer and an evader. These approaches have been successfully applied in the domain of physical security; for instance, the ARMOR system at LAX and the scheduling of Federal Air Marshals both used Stackelberg equilibria [6], [7]. However, a survey of the related literature for the domain of cybersecurity observes that it remains difficult to identify successful implementations of game-theoretic concepts in cybersecurity, in contrast to physical security [7], and a more recent systematic review similarly finds that, despite a large body of theoretical work, game-theoretic approaches are scarcely utilized by cybersecurity practitioners in practice [8]. Nevertheless, despite the apparent practicality of the established game models, there is still an issue of utilizing them in practice.

While the majority of the models developed by researchers and scientists are mathematically sound, the actual application of game theory in practice proved to be a much more challenging endeavor. In particular, the majority of the solutions and recommendations offered were based on the minimax algorithm and the state space of a chess desk – in other words, researchers would calculate the defense strategy that theoretically was most resilient, rather than attempt to apply the game theory in real-time and work with existing network telemetry, operational logs, and standardized threat intelligence feeds.

C. *Traditional Intrusion Detection & Alert Fatigue:*

Commercial enterprise security software relies predominantly on signature-based Intrusion Detection Systems (IDS) and anomaly detection-based Endpoint Detection and Response (EDR). Such methods conventionally trigger alerts upon the execution of malicious artifacts on a compromised end-host, thereby detecting incidents as standalone events.

Consequently, the Security Operations Center (SOC) is deluged with tens of thousands of repetitive event alerts. Empirical measurement of a real-world SOC's alert logs over four years found analysts receiving 24,000 – 134,000 alerts per day, of which only about 0.01 % were determined to be true positives [9]. Analysts are then forced to triage each alert as it comes in, which places tremendous pressure on the SOC team. This procedural bottleneck grants attackers' ample opportunity to pivot laterally within the network infrastructure, often remaining undetected until the point of data exfiltration.

D. *Techniques of Chess Applicable to Cybersecurity:*

Chess provides time-tested strategic and tactical concepts that can be used to address discovered challenges. First, in chess, each piece has its value which is determined by its power and importance on the board. Similarly, network assets can be prioritized according to their criticality to the business. For example, the Domain Controller can be compared to the King, Queen, or Rook in chess, while the Endpoints are similar to Pawns. Second, grandmasters apply various strategies and techniques to secure their position on the board and limit the opponent's play. Such strategies include controlling the center, files, and diagonals, as well as blocking the enemy's movements. These principles can be applied to the cyber-risk domain by using appropriate strategies to secure the network infrastructure and limit the potential attack surface. Finally, minimax with Alpha-Beta Pruning can help determine the best move for the defender by looking ahead 3-5 moves in the game and focusing only on the most promising options while eliminating the least promising lines of defense. The chess analogy would involve using honeypots and bait credentials to lure and trap the attacker.

E. *Research Gap:*

Unlike existing research into attack graph modeling and theoretical game-play analysis, this paper addresses the gap in the existing literature, which lies in developing a real-time deterministic framework that uses enterprise threat intelligence to determine multi-stage lateral movement paths within an enterprise as movements inside a depth-limited chess search tree.

Attack graphs in the literature are too static to reason about ongoing compromises, game-theoretic approaches are too abstract to be applied to real-time telemetry, and existing intrusion detection systems (IDS) are not capable of considering events in a connected way.

The present study addresses this issue by implementing a functional Minimax algorithm with Alpha-Beta pruning, which incorporates the attack surface of enterprise infrastructure as a chess board with weighted moves.

Mapping attacker behaviors' described in MITRE ATT&CK to moves on the board allows the suggested framework to effectively predict potential movement patterns, reducing the time to containment and removing noise from unscored events.

III. PROPOSED SYSTEM AND METHODOLOGY:

This section describes the proposed system.
The designed platform is an innovative threat simulation and network defense mechanism that enables the predictive modeling of multi-stage cyberattacks. The software's main advantage lies in its ability to convert complex real-world enterprise networks into simplified state graphs while aligning potential adversary actions to known frameworks, such as MITRE ATT&CK. This approach allows the system to operate as a threat-hunting aid that analyzes system logs for signs of compromise and uses graph-search heuristics to prioritize the most mission-critical attack paths, surfacing a recommended response (e.g., network segmentation) for a defender to act on; the current prototype recommends such actions as text and does not itself enforce them (Section III-A).

A. *System Architecture:*

The architecture under consideration aims to transform the input data stream of network telemetry into containment recommendations via a three-stage pipeline: strategic game tree search engine, XAI threat mapping module, and automated response interface.

Component 1: ***The AI Minimax Engine***: The core of the analysis is the Minimax search algorithm with a fixed search depth and Alpha-Beta Pruning as the algorithm's optimization layer, following the classical approach to adversarial game search [10], [11]; the evaluation function itself, rather than the search procedure, is hand-built for this domain, in the spirit of early chess engines [12]. The algorithm takes the current state of the enterprise network, represented by a weighted graph G = (V, E, W), and the number of steps ahead for analysis (d = 3, the only depth implemented and evaluated in this work) as input. All possible adversary maneuvres' are scored, and pruning is performed with classical alpha-beta bounds [11], which remove branches that cannot influence the final outcome. Because the full candidate pool is too large to expand without restriction at every ply, only the highest-scoring candidates by E(s) are expanded at each attacker ply (a branching limit of eight in the evaluated implementation), rather than every unvisited technique; no separate ATT&CK-technique-frequency weighting is used. As an output, the algorithm returns the numerical security-risk assessment and the best continuation of the adversary's moves found within this restricted search. Each node's evaluation score is computed as a weighted sum: $E(s) = \Sigma$ (w_asset × asset_criticality) + (w_priv × privilege_level_gained) − (w_dist × distance_to_target), where w_asset, w_priv, and w_dist are tunable weights and asset_criticality reflects the chess-piece value assigned in Section II.D.

Component 2: ***XAI Threat Translator Module***: The XAI threat translator module is the interpretability component of the design, which transforms the raw output of the evaluation engine (e.g., score, primary move trees, and weights) into intuitive human-readable threat models and contextual security indicators. By associating predicted state transitions with MITRE ATT&CK tactics, techniques, and procedures, it enables the system to render its predicted next technique and tactic as a natural-language risk statement and a suggested response action; it does not itself detect zero-day attack paths, and its output has not been evaluated for accuracy independently of the engine's own prediction (Section V-D).

Component 3: ***The User Interface & SOAR Response Hub*****:** Built with an operational focus in mind to accelerate response and reduce mean time to resolution, this capability guides the defender through the attack graph visualized in the form of a proposed attack path. It highlights critical assets (e.g., Domain Controllers, key databases) and potential lateral movement paths for improved situational awareness. In the evaluated prototype, this recommendation is produced by a static lookup from the predicted tactic to a pre-written action string (e.g., "immediate isolation; initiate incident response" for Impact); the hub surfaces this text to a defender and does not itself execute any containment action, such as dropping network routes or terminating processes (Section V-D).

"Unlike chess, the defender cannot observe the true state of an attacker and must instead assume potential positions based on logs/alerts/network traces visible within an observation window. This study discusses the search tree being effectively performed over the belief state of where the attacker might be, given the delayed/incomplete information the defender has access to, which is not present in traditional chess."

B. *Testbed Design and Data Generation Pipeline*

To evaluate the proposed system against realistic; multi-step adversary actions, we constructed a controlled attack-simulation testbed instead of relying on existing public datasets, which, although potentially rich in telemetry, are either outdated, lack

process-level insight, or are unlabeled with respect to particular MITRE ATT&CK techniques of interest. In this section, we describe the architecture of our testbed, adversary emulation protocol, and data pipeline that transforms endpoint telemetry into ATT&CK-labeled feature data for training and evaluation of the Minimax-based evaluation function described in Section III-A.

1) *Testbed Architecture*

Our testbed consists of three isolated virtual machines interconnected on a dedicated internal network segment, logically and physically separated from any production or external-facing network to enable consistent experimental results and prevent undesirable side effects: (i) a Windows endpoint instrumented with Sysmon for high-level fidelity telemetry capture, representing a target vertex $v \in V$ in the graph $G = (V, E, W)$; (ii) an adversary-emulation host running the automated red-team platform; and (iii) a log-collection host that hosts telemetry aggregation, parsing, and feature extraction processes. The division between endpoint instrumentation, attack simulation, and log processing follows operational best practices in real-world enterprise security operations, where endpoint monitoring and attack defense capabilities are distinct from log collection and parsing.

2) *Adversary Emulation*

Rather than manually authoring attack techniques, our testbed is configured with the MITRE Caldera adversary emulation platform to execute reproducible multi-step attack chains. Caldera is appropriate for this purpose since each of its configurable adversary "abilities" are already pre-mapped to a corresponding MITRE ATT&CK technique identifier in its ability definition schema. This is a methodologically important distinction, as it ensures that ground-truth technique labels are derived directly from the emulation platform's capability metadata rather than requiring manual pattern-matching heuristics over raw command-line telemetry, which could suffer from false negatives (omitted technique variants) and false positives (similar-looking command-line syntax across unrelated techniques). Caldera's ability-based approach provides an accurate and automated way to ground-truth ATT&CK techniques executed by our simulated adversary, and is elaborated upon in Section III.C.

Caldera's ability definitions were executed against the Windows endpoint in the form of a remote-access agent to establish a command-and-control channel through which adversary operations — organized into named "operations" against a given "adversary profile" (e.g., a Discovery-tactic profile consisting of abilities like local user enumeration, process discovery, network share discovery, and security-software discovery) — were executed and their success/failure recorded.

3) *Endpoint Telemetry Collection*

Sysmon (System Monitor), a Windows system service and device driver that writes detailed system logs to the Windows Event Log, was instrumented with a community-maintained configuration baseline to reduce noise while enabling rich coverage of event categories relevant to security analysis. Sysmon was selected for its industry adoption, minimal performance overhead, and comprehensive per-event schema that includes process creation, parent-child process lineage, and full command-line arguments, which directly map onto the privilege_level_gained and distance_to_target terms of the evaluation function E(s) described in Section III-A: each ability executed by Caldera manifests itself as one or more process-creation events in the endpoint's event log trace, which can be associated with the originating ATT&CK technique.

4) *Data Extraction Pipeline*

Following each operation in the defender's attack-simulation playbook, endpoint event logs were exported in Windows Event Log (.evtx) binary format and uploaded to the log-collection host over an internal HTTP channel. The binary event logs were parsed into structured, flat tabular records using python-evtx, extracting relevant fields such as event timestamp, event type, process image path, parent process image path, command-line arguments, and user context. This produces a per-event log dataset ready for downstream technique labeling and feature engineering.

5) *Ground-Truth Technique Labeling*

In order to associate parsed telemetry events with their corresponding ATT&CK techniques, we programmatically extracted adversary-emulation platform capability definitions to produce a lookup table that maps each capability's identifier, tactic category, ATT&CK technique ID and name, and command syntax per-platform. Sysmon-observed command-line values were matched to ground-truth entries in this capability lookup table using pattern and temporal proximity to the operation window to enable labeling of observed low-level system events with their corresponding high-level adversary technique without requiring manual signature definition. This technique labeling approach underpins the ATT&CK-alignment assertion of the XAI Threat Translator Module described in Section III-A, and provides the technique-frequency weighting data used in the Minimax branch-ordering heuristic (Section III-A)

IV. RESULTS

A. Data Pipeline Output

The testbed pipeline was validated end-to-end on the thesis-op-1 Caldera Discovery operation executed against Endpoint-1. Sysmon telemetry export yielded 3,855 total process-creation events across the operation window. Filtering to PowerShell/cmd-driven events (the executors used by the deployed abilities) reduced this to 32 events, comprising 12 unique command-line invocations.

Cross-referencing these command lines against the ground-truth ability-to-technique mapping extracted from Caldera's own Stockpile plugin (323 ability definitions, 179 applicable to the Windows platform) via token-overlap matching yielded 8 events labeled with a specific ATT&CK technique ID, and 4

events correctly identified and excluded as non-attack artifacts (bare agent-launcher invocations and an empty-payload beacon check-in, verified by manual inspection rather than accepted at face value from the automated matcher).

TABLE I. GROUND-TRUTH-LABELED BELIEF-STATE TRACE (THESIS-OP-1)

Time (UTC)	Technique ID	Ability	Tactic	Match Basis
07:22:03.439	T1087.001	Identify local users	Discovery	Token overlap
07:22:49.164	T1057	Discover injectable process	Discovery	Token overlap
07:22:56.716	T1057	Discover injectable process	Discovery	Token overlap
07:24:27.716	T1135	View admin shares	Discovery	Manual (keyword)
07:25:17.148	T1018	Discover domain controller	Discovery	Token overlap
07:25:51.805	T1518.001	Discover antivirus programs	Discovery	Token overlap
07:26:44.712	T1615	Discover Group Policy Info	Discovery	Manual (keyword)
07:27:37.050	T1518.001	Identify Firewalls	Discovery	Token overlap

Two of the eight labels (T1135, T1615) required manual correction after the automated token-overlap matcher mis-assigned them to a higher-token-overlap but semantically unrelated generic ability; this is discussed as a limitation of pure lexical matching in Section V.

B. Evaluation Function Behavior

The evaluation function E(s) = w_asset × asset_criticality + w_priv × privilege_level_gained − w_dist × distance_to_target was applied uniformly with unit weights (w_asset = w_priv = w_dist = 1.0), using MITRE ATT&CK tactic category as a proxy for both asset_criticality and privilege_level_gained (Table II).

TABLE II. TACTIC-DERIVED SCORING WEIGHTS

Tactic	Asset Criticality	Privilege Gain
Discovery	3	0
Collection	5	1
Execution	5	2
Persistence	6	2
Defense Evasion	4	2
Credential Access	7	3
Command and Control	6	3
Lateral Movement	8	4
Exfiltration	9	4
Privilege Escalation	9	5
Impact	10	6

C. Prediction and Comparative Scoring

The current inferred attacker state at the end of the trace was T1518.001 (Discovery tactic). All eight observed states scored comparatively low under this weighting, since Discovery contributes the lowest asset-criticality and privilege-gain values in the table — for example, the T1018 (Discover domain controller) event, at estimated distance 2, scored E(s) = 3 + 0 − 2 = 1.0.

Searching the full 323-technique candidate pool for the highest-value technique not already present in the trace, the engine predicted T1485 (Clear Current User Recycling Bin), an Impact-tactic technique, at estimated distance 3: E(s) = 10 + 6 − 3 = 13.0.

The twelve-point gap between the highest-scoring in-trace state (T1018, E(s)=1.0) and the predicted next state (T1485, E(s)=13.0) reflects the formula correctly discriminating between low-value reconnaissance actions and a high-value escalation candidate, rather than simply selecting the most recently-seen tactic. This is consistent with the intended belief-state escalation logic: because the observed trace consists entirely of Discovery-tactic actions, the model favors transitioning toward a higher-weighted tactic category as the next likely move.

The XAI Translator module rendered this prediction as a natural-language statement identifying the predicted technique and its associated tactic. The SOAR stub correctly mapped the Impact tactic to an "immediate isolation; initiate incident response" recommended action, versus the "increase monitoring; no containment action needed" response that would have been generated had the top prediction remained within the Discovery tactic.

Two caveats apply to this analysis. First, the prediction above comes from the single-ply configuration of the engine; the depth-3 search and its evaluation are reported in Sections IV-E and IV-F. Second, as Section IV-E shows, the same preference for high-value candidates caused the engine to return T1485 at most prediction steps regardless of the observed history, so the behavior described here demonstrates severity ranking of candidate techniques rather than trajectory-aware prediction.

D. Extended Dataset (Operations 2 and 3)

Two further Caldera operations were run against Endpoint-1 with the same configuration as thesis-op-1 (group red, atomic planner, plain-text obfuscator): a Collection operation (four abilities) and a Defense Evasion operation (sixteen abilities across several tactics). A third operation using a lateral-movement adversary profile was attempted and abandoned. Most abilities in that profile require a second target host, which the single-endpoint testbed lacks, and pointing the remote-host fact at Endpoint-1 itself produced only two executed links.

Telemetry for both operations was exported from the Sysmon Operational log of Endpoint-1 in a single pass and processed with the pipeline of Section IV-A. Two data-handling problems were found and corrected. An initial export was served from a stale file left over from thesis-op-1 and reproduced that operation's 3,855 events exactly, so it was discarded and the log was re-exported. In addition, the export contains the full cumulative Sysmon history rather than only new events, so it also held earlier activity, including the thesis-op-1 run of 30 August and events labeled T1029 (Scheduled Exfiltration) on 3 and 6 September whose origin was not investigated. Only events dated 26 September 2026 (UTC) or later, the day of the two operations, were retained, which reduced the labeled events from 67 to 46.

Consecutive occurrences of the same technique were collapsed into one, which reduced thesis-op-1 from eight to seven steps and the retained op-2/op-3 events from 46 to a 30-step sequence (Table III). The most frequent repeat, T1059.001 ("Emulate Administrator Tasks"), recurs at short intervals throughout the session. Because the token-overlap matcher may be assigning generic PowerShell launches to this one broad ability, it was treated as background execution activity and only immediate repeats were merged; it still accounts for 11 of the 30 op-2/op-3 steps. The op-2/op-3 sequence spans eight tactic categories (collection, execution, defense evasion, impact, persistence, credential access, discovery and exfiltration). It also contains four service-creation steps (T1569.002) from the abandoned lateral-movement attempt, which ran in the same session, so it is a mixed single-session trace rather than a record of operations 2 and 3 alone.

TABLE III. SUMMARY OF THE TWO EVALUATED DATASETS

Quantity	thesis-op-1	op-2/op-3 (cumulative export)
Sysmon events parsed	3,855	20,393
PowerShell/cmd events	32	185
Unique command lines	12	64
Events with a technique label	8	67 (46 on or after 26 Sep)
Sequence steps after de-duplication	7	30
Prediction steps	6	29

E. Depth-3 Minimax Engine and Static Baseline

The single-ply selection of Section IV-C was extended to a depth-3 alpha-beta minimax search over attacker, defender and attacker plies. At attacker plies the engine expands candidate techniques not yet observed and scores each with E(s), using the weights of Table II and a fixed distance estimate of 3. At the defender ply the defender either takes no action or applies a containment response, modeled as a fixed penalty of 5.0 subtracted from the attacker's continuation value, and chooses whichever is worse for the attacker. The candidate pool (unique technique IDs across the 323 ability definitions) is too large for an unrestricted depth-3 search, so only the eight highest-scoring candidates were expanded at each attacker ply.

A first run with the tactic-only E(s) returned T1485 (Impact) at all 35 prediction steps regardless of history, because the highest-value tactic always ranks first. A kill-chain adjacency bonus of 8.0 was then added for candidates whose tactic equals, or immediately follows, the tactic of the latest observation in a fixed order (discovery, collection, credential access, privilege escalation, lateral movement, persistence, defense evasion, command and control, exfiltration, impact). This change was made after seeing the first results and was not checked on held-out data. The constants (bonus 8.0, defender penalty 5.0, branching limit 8) were set by hand and were not adjusted after the comparison below. With the bonus, the prediction differed from T1485 at only six of the 35 steps, all in the op-2/op-3 sequence.

The comparator is a static rule with no scoring or lookahead. It takes the tactic of the latest observation, moves to the next tactic in the same fixed order (or to discovery if the latest tactic is not in the order, as is the case for execution), and predicts the first unobserved technique of that tactic in alphabetical order of technique ID. It stands in for rule-based tooling, which has no built-in next-step prediction, and does not model any specific commercial IDS. Both systems were evaluated in a walk-forward manner: at each position of a sequence, they saw only the history up to that point and predicted the next technique, giving 35 prediction steps (6 in thesis-op-1 and 29 in op-2/op-3). Acc_K is the fraction of steps predicted correctly, computed at the level of the exact technique ID and, as a coarser measure, at the level of the tactic.

F. Prediction Accuracy and Hypothesis Test

Table IV gives the results. Neither system predicted the exact next technique at any of the 35 steps. At the tactic level each system was correct at one step (1/35 = 0.029), and the two steps were different ones, both in the op-2/op-3 sequence. Paired

McNemar tests [13] on the pooled steps gave p = 1.000 for exact technique ID (no discordant pairs) and p = 0.4795 for tactic level (b = 1, c = 1). The null hypothesis $H_{0,1}$ is therefore not rejected: the data give no evidence that the depth-3 minimax predicts the next technique better than the static rule. With these few steps they also cannot show that the two are equivalent.

TABLE IV. NEXT-STEP PREDICTION ACCURACY AND PAIRED McNEMAR TESTS

Metric	Sequence	Steps	Minimax	Baseline	McNemar (pooled)
Exact technique ID	thesis-op-1	6	0.000 (0/6)	0.000 (0/6)	—
Exact technique ID	op-2/op-3	29	0.000 (0/29)	0.000 (0/29)	—
Exact technique ID	**Pooled**	**35**	**0.000 (0/35)**	**0.000 (0/35)**	**p = 1.000 (b = 0, c = 0)**
Tactic level	thesis-op-1	6	0.000 (0/6)	0.000 (0/6)	—
Tactic level	op-2/op-3	29	0.034 (1/29)	0.034 (1/29)	—
Tactic level	**Pooled**	**35**	**0.029 (1/35)**	**0.029 (1/35)**	**p = 0.4795 (b = 1, c = 1)**

Two features of the protocol limit these numbers. Both systems exclude already observed techniques from their candidate sets, but 18 of the 35 prediction targets (17 of 29 in op-2/op-3 and 1 of 6 in thesis-op-1) repeated a technique already in the history. Exact-ID accuracy therefore could not exceed 17/35 = 0.486 for either system, even with a perfect ranking. In addition, the discordant-pair counts (0 and 2) are too small for the chi-square approximation to be reliable. An exact binomial McNemar test gives p = 1.000 in both cases, so the conclusion is unchanged.

V. DISCUSSION AND LIMITATIONS

A. Interpretation of the Hypothesis Test

H_1 predicted that a depth-3 minimax lookahead would predict the next attacker step significantly more accurately than a static rule-based approach. The results do not support it: the two systems scored identically at both levels of evaluation, and the paired tests show no significant difference (Section IV-F). This is a failure to find support in a small pilot and is not evidence that lookahead cannot help. With 35 paired steps from two sequences and both systems near zero, the test had little power to detect even a large difference. The steps are also not independent, since consecutive steps within a sequence share history, so the reported p-values, if anything, overstate the available evidence. The case for proactive prediction over reactive alerting remains an argument for the approach; this dataset does not demonstrate it.

In summary, the paper does not present strong evidence for its central claim, and three claims should be kept apart. The claim that a depth-3 minimax predicts the next technique more accurately than a static rule is not supported. The broader premise that lookahead allows earlier and more useful response than reactive alerting is not tested by these data. What the results do support is narrower: an end-to-end testbed that produces ground-truth-labeled telemetry, together with a paired evaluation harness, can be built and run, and a tactic-scored lookahead without transition information gives mostly constant predictions. Stronger evidence would need a larger corpus and a stronger baseline (Section V-E).

B. Limitations of the Engine Design

All scoring inputs are hand-set: the unit weights, the tactic-derived asset-criticality and privilege values of Table II, the defender penalty, the adjacency bonus and the branching limit. No sensitivity analysis was run, and the adjacency bonus was introduced after the first results were seen. Neither system uses empirical technique-to-technique transition statistics; their only knowledge of ordering is the fixed kill-chain list. Against a candidate pool this large, exact-ID prediction is close to guessing for both.

The design also has specific weaknesses visible in the results. The engine returned T1485 at 29 of 35 steps even with the adjacency bonus. By construction, the same-tactic part of the bonus lets a hypothetical Impact move be followed by another Impact move at a bonus in the lookahead, which reinforces the highest-value tactic. Execution is missing from the fixed tactic order and was the latest observed tactic at 16 of the 35 steps, so the bonus did not contribute at the first ply at those steps, and the baseline fell back to discovery. Finally, both systems exclude techniques already observed, which rules out the repeated techniques that make up about half of the prediction targets (Section IV-F). These issues confound the comparison, so the results say little about lookahead as such.

C. Limitations of the Data

The evaluation uses two sequences from one lab, one endpoint and one operator. The traces come from Caldera's scripted adversary emulation rather than from a human or adaptive attacker, so the task is closer to predicting Caldera's ability scheduling than real attacker behavior. Ground-truth labels come from token-overlap matching of command lines against ability definitions, with rule-based corrections. Two of the eight thesis-op-1 labels (T1135 and T1615) needed manual correction, and the op-2/op-3 labels were not systematically audited, so further mismatches are likely. The recurring T1059.001 label is a particular concern, as noted in Section IV-D. More broadly, evaluating a detection approach on a small corpus collected in one lab is exactly the kind of setup that established cautions about machine-learning-based intrusion-detection evaluation warn can fail to generalize [14].

The retained session window mixes activity from different sources. It includes service-creation steps from the abandoned

lateral-movement attempt, and a T1029 (Scheduled Exfiltration) label appears once inside the window, in addition to the two earlier occurrences removed by the date filter; its origin was not investigated. The date filter works at whole-day resolution and Sysmon exports are cumulative, so isolating single operations cleanly would need a log clear or a time-bounded export before each run.

D. Limitations of the Testbed

The testbed has one Windows endpoint, no domain controller and no endpoint protection product, and the lateral-movement operation had to be dropped. As a result, the tactics with the highest weights in Table II (privilege escalation and lateral movement) do not appear in either sequence, and the sequences consist mostly of low-weight tactics. How the weighting scheme behaves on richer attack chains remains untested. The XAI and SOAR outputs were also not evaluated quantitatively in this work.

E. Future Work

Future work should use a larger and more varied corpus (several hosts, longer and repeated operations, and sequences containing higher-weight tactics), learn technique-transition statistics from data, allow repeated techniques and handle execution properly, and report how sensitive the results are to the scoring constants. Stronger baselines, such as a first-order Markov model over techniques, would give a fairer comparison than a fixed kill-chain rule. Hyperparameters should be fixed before evaluation and exact tests used. Measuring early-warning value, for example how many steps ahead a system flags a high-impact technique, would also fit the proactive-detection goal better than exact next-step matching.

VI. CONCLUSION:

This paper described a testbed and pipeline that turn emulated attacks into ATT&CK-labeled sequences, a depth-3 alpha-beta minimax engine that predicts the next technique, and a static kill-chain baseline, and used them to test whether lookahead improves next-step prediction. On 35 prediction steps drawn from two sequences the two systems performed identically: neither predicted the exact technique at any step, and each was right at the tactic level once. The paired tests show no significant difference, so H_1 is not supported by this dataset.

The pilot does not show that lookahead helps, and it does not show that it fails to help. The small sample, the hand-set scoring, the absence of transition information and the mixed session data limit what can be concluded, and the engine mostly returned a single high-value technique. What the work does provide is a working pipeline, an evaluation protocol that can be reused with more data, and a record of the design and data problems to avoid. The next steps are a larger and more varied corpus, transition statistics learned from data, stronger baselines and hyperparameters fixed in advance (Section V-E).

VII. REFERENCES:

[1] MITRE Corporation, “Caldera: A scalable, automated adversary emulation platform,” software. [Online]. Available: https://github.com/mitre/caldera

[2] Microsoft, “Sysmon,” Sysinternals documentation, Microsoft Learn.

[3] B. E. Strom, A. Applebaum, D. P. Miller, K. C. Nickels, A. G. Pennington, and C. B. Thomas, “MITRE ATT&CK: Design and philosophy,” MITRE Corporation, Technical Report, 2018.

[4] X. Ou, S. Govindavajhala, and A. W. Appel, “MulVAL: A logic-based network security analyzer,” in Proc. 14th USENIX Security Symposium, 2005, pp. 113–128.

[5] X. Ou, W. F. Boyer, and M. A. McQueen, “A scalable approach to attack graph generation,” in Proc. 13th ACM Conf. on Computer and Communications Security (CCS), 2006, pp. 336–345.

[6] M. Tambe, Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned. Cambridge, U.K.: Cambridge University Press, 2011.

[7] J. Pawlick, E. Colbert, and Q. Zhu, “A game-theoretic taxonomy and survey of defensive deception for cybersecurity and privacy,” ACM Computing Surveys, vol. 52, no. 4, pp. 1–28, 2019.

[8] B. Collins, S. Xu, and P. N. Brown, “Game-theoretic cybersecurity: The good, the bad and the ugly,” arXiv preprint arXiv:2401.13815, 2024.

[9] L. Yang, Z. Chen, C. Wang, Z. Zhang, S. Booma, P. Cao, C. Adam, A. Withers, Z. Kalbarczyk, R. K. Iyer, and G. Wang, “True attacks, attack attempts, or benign triggers? An empirical measurement of network alerts in a security operations center,” in Proc. 33rd USENIX Security Symposium, 2024, pp. 1525–1542.

[10] C. E. Shannon, “Programming a computer for playing chess,” Philosophical Magazine, ser. 7, vol. 41, no. 314, pp. 256–275, 1950.

[11] D. E. Knuth and R. W. Moore, “An analysis of alpha-beta pruning,” Artificial Intelligence, vol. 6, no. 4, pp. 293–326, 1975.

[12] M. Campbell, A. J. Hoane Jr., and F.-h. Hsu, “Deep Blue,” Artificial Intelligence, vol. 134, no. 1–2, pp. 57–83, 2002.

[13] Q. McNemar, “Note on the sampling error of the difference between correlated proportions or percentages,” Psychometrika, vol. 12, no. 2, pp. 153–157, 1947.

[14] R. Sommer and V. Paxson, “Outside the closed world: On using machine learning for network intrusion detection,” in Proc. IEEE Symposium on Security and Privacy, 2010.