HAZARDOUS SOFTWARE

Hazardous Software Vision of Serious Gaming for Strategic Computer Network Operations

Serious gaming offers many opportunities with respect to cyber operations over traditional modeling and simulation. Hazardous Software's vision for serious gaming encompasses several factors: 1) characterizing the agents and actions within the scenario; 2) focusing on usability and deployability, applying user-based time manipulation over the game to let users explore the strategy space across alternatives; 3) using appropriate utility-based modeling; 4) leveraging existing policy and process frameworks, enabling users to create new policies and processes, and learn how they interact; and 5) looking at agents' goals, motives, decision processes, and strategy via a game theoretic approach.

Agents and Actions Choosing proper agents and actions are paramount for creating an effective serious game. Strategies within the cyber domain are often adversarial and rarely isolated from politics. To understand the cyber landscape, players would benefit from competing in opposing roles on offense, defense, and network exploitation, combined with playing as different entities such as countries and organizations which include differing political, economic, and kinetic implications for their actions. Strategic-level serious games should also perform the mundane and repetitive aspects of a scenario for the player in the same way that mundane and repetitive IT tasks, business processes, and attacks should be automated. A player that is continuously engaged with the game will be more receptive to learning what the game is teaching. Rather than focusing all of a player's effort on making specific decisions, a strategy-level serious game should also focus on policies and the decision process itself. Direct representation of the decision process can be an instructive way to introduce new leaders to their roles and to allow key decision makers to focus on anomalous incidents while automating the common.

Usability and Deployability A distinguishing characteristic of serious games from traditional modeling and simulation is the ease of playability and deployability. Simulation tools often require complex configuration and experts for operation, whereas serious games are designed for a wider audience to begin learning and exploring. Other foci of serious games are ease of deployment and real-time feedback to players' decisions. Social networking and crowdsourcing can be useful in serious games to foster communities to create game content and share strategies.

Time Manipulation and Strategy Exploration The ability to move to different positions in time is a fundamental aspect of gaming. Resetting the game, going back to a previous state after losing a "life", or freely jumping around the timeline are all different ways of affording a player the ability to take on risk in a safe environment to foster learning. More complex time manipulation controls that permit players to quickly and easily move around the timeline permit players to explore the long-term implications of decisions and intuitively perform qualitative sensitivity analysis. Understanding the second and third order effects of decisions is important for offensive, defensive, and exploitation matters. Configuration management systems are crucial for managing and merging inputs from many individuals in the development and deployment of information systems. Decision policies are similar to software code; the ability for multiple players to collaboratively and simultaneously edit a strategy across a timeline in real time enables multiple subject matter experts to work in concert collaboratively and adversarially. When playing collaboratively, players can learn how their strategies may conflict and where their strategies may be complementary. When playing adversarially, multiplayer time manipulation helps players explore the strategy space of how their adversaries can thwart plans, create distractions, and exploit weaknesses.

Utility-Based Modeling, Preferences, Goals, and Constraints Cyber security often conflicts with other factors, such as usability, performance, and cost. Being able to defend against a large scale DDoS attack may be worth the costs of redundant equipment, but requiring a complex two-factor authentication for software testers to submit a bug report from within an internal network is unlikely to be worth the cost or effort. Similarly, purchasing a SCADA exploit from a rogue group to use against an adversary may have political risks. Users' and attackers' preferences and goals are important to learning the dynamics of a complex system. An attacker may wish to deny a particular service, extract information, corrupt information, or some combination of those activities. A hacking group that does not report to a government may be largely free from political constraints, whereas a nationally sponsored cyber attack may have severe political implications and constraints, especially when the decision arises of whether to publicly claim such an attack. Users of a network typically wish to perform their jobs effectively and their incentives may not always be aligned with good security practices. A cyber security game that includes the possibilities of organizational policy, politics, operating costs, social engineering, and kinetic retaliation will better prepare players for real-world complexities.

Policy and Process Experimentation Deciding how to react to threats is a multifaceted problem. How many resources should be spent investing in stronger protocols? How effective is user education and what kind of ROI does security education yield and how does it compare to securing systems? Are honeypots worth deploying

when faced with a particular threat? If allowed by ethical, legal, and political conditions, is taking an offensive position against an advanced persistent threat justified, useful, and will it do any good? A strategy-level cyber defense serious game should help users think about these complex trade-offs and ascertain how different choices can yield synergy or unintended consequences. Further, a strategy-level cyber defense game that can intuitively help users combine existing policies and processes, create new ones, and evaluate their effectiveness will provide a better contextual whiteboard from which to collaborate and innovate to find the appropriate course of action.

Game Theoretic Approach to Strategy and Decision Processes A significant amount of cyber defense and attack work focuses on hard security practices such as authentication (a user is who the user says or that a message is authentic) and detection, but soft security measures such as trust, patience, and context of actions are just as important in preventing exploitation of weaknesses. Reputation and anomaly detection are reactive measures, using statistics and looking in hindsight. Trust and prevention are proactive measures, using game theory and looking to future attacks. In evaluating different preventative defensive measures, a player may be offered choices between different levels of security, organizational effectiveness, and cost. For networks containing low-risk data and services, an expected value may be employed when evaluating strategies; costly but improbable attacks may not be given any special treatment over frequent attacks that are simply annoyances to daily operations. Conversely, for networks containing high-risk data and services, a minimum regret or minmax approach may be necessary to focus on and prevent a highly improbable but extremely damaging attack. Determining proper redundancy, ambiguity, and security of services over time with respect to a multitude of attackers and budgetary constraints are important strategic concepts to convey to the player.

Hazardous Software is uniquely positioned in the market to meet the needs of strategy level serious gaming, particularly in the domain of cyber defense.

Hazardous Software's Resequence Game Engine is the only game engine on the market positioned to teach causality and long-term effects by free-form multiplayer timeline manipulation. Because the players can play at any point on the timeline in fast or slow motion, they must constantly evaluate and reevaluate the consequences of each of their decisions. Players can issue or retract commands in the past, present, and future, blurring the boundary between hypothetical and committed decisions. In particular, the player can revisit critical decision points, learn what decisions had the most long-term impact, and use statistical information on the timeline to make more informed decisions. Resequence's support for multiple players to simultaneously change history can further enhance training. Each player can take advantage of an opponent's strategic weaknesses in the past, and each player can correct mistakes in strategy and determine the best response to their opponent. This helps players to find minmax strategies. Loss of information asymmetry can be severely detrimental to a given strategy, but Resequence pushes players to account for the possibility that their opponent may learn of their strategy before it is executed. Players can look into the future to see what the outcomes of their current strategies and their opponent's strategies will be, giving the players the ability to analyze the long-term effects of decisions. Multiple players can collaboratively plan using information on the timeline to prevent conflicting plans. Folding information from future points in the simulation back to previous states enables a "Newton's Method" technique for converging on strategies.

Amalgam is a game engine, programming language, and tool platform currently under development by Hazardous Software that gives developers and players the ability to easily combine objects, rules, logic, and behaviors to combine and easily create new game mechanics and game objects. Developers and players can build new mechanics and objects by mashing together entities that exhibit the desired qualities without having to write source code. These mechanics and objects can be shared to create new games and new experiences within existing games. Amalgam also features an integrated rules and permissions framework to give players flexibility in creating custom game content without breaking the general rules of the game they are playing in. The engine is currently under development, initially on the JVM supporting 2D games, and will soon be ported to C++ and mobile devices, as well as longer-term plans to integrate with the Hazardous Software time manipulation game engine Resequence.

Further, Hazardous Software has developed many tools to analyze and design games via utility and game theory. By employing modern game theory techniques, Hazardous Software has techniques to measure whether the game is teaching the strategies intended and to ensure balanced gameplay.