APT RPG: Design of a Gamified Attacker/Defender Meta Model

Robert Luh[1,2,3], Marlies Temper[1], Simon Tjoa[1,2] and Sebastian Schrittwieser[1,2]
[1]*St. Pölten University of Applied Sciences, St. Pölten, Austria*
[2]*Josef Ressel Center TARGET, St. Pölten, Austria*
[3]*De Montfort University, Leicester, U.K.*
{*first.last*}*@fhstp.ac.at*

Keywords: security model; gamification; attack patterns; controls; malware; intrusion detection

Abstract: We present a meta model for comprehensive, time-enabled attacker/defender behavior ready for incorporation in a dynamic, imperfect information multi-player game that derives significant parts of its ruleset from established information security sources such as STIX, CAPEC, CVE/CWE and NIST SP800-53. Concrete attack patterns, vulnerabilities, and mitigating controls are mapped to their counterpart strategies and actions through practical, data-centric mechanisms. The gamified model furthermore considers and defines a wide range of actors, assets, and actions, thereby enabling a detailed assessment of cyber risks while giving analysts the opportunity to explore specific attack scenarios in the context of their own infrastructure.

1 Introduction

IT systems are threatened by a growing number of cyber-attacks. With the emergence of Advanced Persistent Threats (APTs), the focus shifted from off-the-shelf malware to multipartite attacks that are tailored to one specific entity. These targeted threats are driven by varying motivations, such as espionage or high-profile sabotage, and often cause significantly more damage.

APTs are typically conducted by dedicated groups within organized crime, industry, or nation state intelligence and increasingly affect less prominent targets as well. While APTs utilize malware like most other, more conventional attacks, their level of complexity and sophistication is usually significantly higher. It is becoming increasingly difficult to plan and implement an organization's defense in response to the targeted threat. At the same time, attack patterns, vulnerabilities, and mitigation control catalogs provide a information that can rarely be interleaved due to varying levels of granularity or abstraction. Most importantly, few models provide a means to map concrete system events as seen on affected assets to a semantic interpretation, known system flaw or definitive countermeasure. Time and system interdependencies are rarely considered. The multi-stage nature of emerging APTs makes it even harder for analysts or business actors to address current threats from the cyber domain. This increasing complexity makes it vital to explore novel approaches to attack/defense modeling, threat intelligence, knowledge extraction, and malicious activity detection on multiple layers, while preserving the flexibility needed to model new scenarios.

The gamified model introduced in this paper helps to assess the risk of targeted attacks on arbitrary infrastructures by providing an extensible framework for simulating attacker and defender behavior in an adversarial setting. APT campaigns of various scope and impact can be modeled on different levels of abstraction, while incorporating the temporal factor in addition to systemic provider/consumer relationships. Even though the model has been created with maximum flexibility in mind, the introduced rule set also provides a concrete solution for depicting and assessing real-life cyber-attacks on private, corporate, and national IT infrastructures. All the required interacting components are based on established languages and standards for observables, actors, attack patterns, and defense measures. Means to mapping on-system events such as sequences of system operations, function calls or kernel operations to the game model are an integral part of the RPG and can be linked to novel and existing pattern or anomaly detection systems.

Furthermore, the game based on the introduced model helps to generate new knowledge and to understand how specific threats impact a system. Shortcomings in current defense implementations are inadvertently highlighted. Managers are encouraged to explore new threats and are presented with possible

organizational and system-level mitigations as suggested by accepted security standards. Analysts and IT/security experts are provided the means to model concrete hacks and link them to the output of intrusion detection systems currently in place at their organization. Altogether, the gamified approach is designed to raise awareness and improve risk assessment while closing the gap between high-level kill chains, independently published countermeasures and policies, and concrete attacks depicted by real-word events and carried out through the exploitation of systemic vulnerabilities and weaknesses.

Specifically, we contribute by:

- presenting an easily expandable, time-enabled attacker/defender meta model for depicting and assessing advanced persistent threats;
- providing a link to various standards and formats, such as STIX-defined data observables, CAPEC attack patterns, as well as operational risk assessment and mitigation planning within a gamified setting;
- introducing a mapping mechanism for linking attacker behavior to opposing security and privacy controls listed in the NIST SP800-53 standard.

Please bear in mind that the game implementation itself is not part of this model-centric paper. It will be discussed and evaluated in an upcoming publication.

The remainder of this paper is structured as follows: After exploring related work in both the intrusion modeling and game theory domains (Section 2), we discuss the theoretical aspects of our model and present a high-level view on the multi-layered approach employed (Section 3). We furthermore define actions and all their component classes in the context of the APT RPG, while paving the way for an accessible game rule system that has been concurrently developed. Section 4 specifies our practical approach to combining STIX vocabularies, CAPEC patterns, CWE, CVE entries, and NIST SP800-53 controls into one model, which takes all its components from existing sources across the threat information domain. Section 5 concludes the paper and provides a research outlook.

2 Related Work

Since the APT RPG borrows concepts from attack modeling as well as game theoretic approaches, we take a closer look at similar works in both areas. Several researchers introduced *attacker/defender models* that consider numerous factors and properties:

Like the APT RPG, the Diamond Model of Intrusion Analysis (Caltagirone et al., 2013) establishes the basic elements of generic intrusion activity, called an event, which is composed of four core features: adversary, infrastructure, capability, and victim. It extends events with a confidence score that can be used to track the reliability of the data source or a specific event. While some of its premises are similar to our own work, the Diamond Model does not consider Enablers or Disablers (see Section 3.3) or any automation for determining specific actions on the attacker's and defender's side. While it is a powerful template in its own regard, the APT RPG provides these mechanisms – and more. Its gamified core provides the ready-to-use framework for simulation and automated knowledge discovery. In summary, the Diamond Model and the APT RPG share commonalities and could potentially benefit from each other in terms of feature modeling, terminology, and approach to gamification.

In the work by Syed et al. (2016), the authors present a unified cyber security ontology (UCO) extending the Intrusion Detection System ontology by Undercoffer et al. (2004). UCO is a semantic version of STIX with a link to cyber security standards similar to the ones used in the APT RPG. Real-world knowledge appended using the Google knowledge graph and various other knowledge bases. Syed et al. provide little information about data retrieval mechanisms and general automation. The main use cases emphasized are the identification of similar software, and the association of vulnerabilities with certain (classes of) products. Unlike the APT RPG, UCO does not consider temporal information or measures of uncertainty.

On the side of *game theory*, introduced models are usually separated into several classes, depending on a multitude of factors. Roy et al. (2010) survey a number of approaches for the network security domain and categorize non-cooperative games into static and dynamic scenarios of varying levels of information quality and completeness. A good starting point for literature review, we classified the APT RPG in accordance to Roy et al.'s schema (see Section 3.2).

Cook et al. (2016) presents an overview of risk assessment strategies for industrial control systems (ICS) that includes, among others, the game theoretic approach. The authors conclude that there exists no unified risk model for hitherto unconsidered ICS scenarios that incorporates events, threats, vulnerabilities, and general consequences with a measure of uncertainty. Suggested future work includes the development of an environment to allow an intelligent adversary to test an ICS's defenses in a nondestruc-

tive (e.g. game theoretic) manner. This is specifically addressed by the APT RPG. Similarly, Lewis (2014) offers an introduction to risk assessment strategies for critical infrastructure protection, including an in-depth look at Bayesian belief networks and game theory applications.

Applied to a power grid scenario, Holmgren et al. (2007) introduce a model for studying defense strategies against hostile actions leading to a power shutdown. Outages are split into stages ranging from prevention to recovery. Unlike the more flexible APT RPG, Holmgren et al.'s model is a perfect and complete information game that does not allow adding new components to the existing network of assets. In addition, only qualified attackers of static skill, determination, and existing access to the system are considered. The overall motivation of implementing the power grid game is still comparable to our approach: It aims to help with resource planning, risk screening, and with studying generic mechanisms that enhance the overall understanding of attacks against a system.

Contrary to the above, Nguyen et al. (2009) formally describe an abstract zero-sum, incomplete information game for network security scenarios. Their network model is based on the concept of linear influence networks (Miura-Ko et al., 2008), which is represented by two weighted directed graphs that signify the relationship of security assets as well as denoting vulnerability correlation among the nodes. The resulting influence matrix describes the contribution of an asset to overall security. If an asset is taken down by the attacker, its node is removed from the network, lowering the security rating for all connected assets. For common assets, compromising one node changes the probability that another linked asset will come under attack. Unlike the APT RPG, Nguyen et al. do not consider specific actors, assets, scenarios, or data sources. Since compromised nodes are removed entirely from the network, the model does not directly support stepped attacks or semi-successful attacker actions. As a pure zero-sum game, the model does not offer the same level of flexible payoff as the system introduced in this paper.

3 Attacker/Defender Model

3.1 Overview

Our gamified approach is based on a novel attacker/defender model tailored to depict interconnected services that maintain – and operate with – various types of information. Consuming and providing services are represented by parent and child entities that enable the modeling of arbitrary, interdependent systems and infrastructures. At the same time, the APT RPG considers on-system events that define nominal and anomalous behavior, thereby establishing a link to actual attacker behavior in the form of individual actions.

Another main advantage of the model is its flexibility: Services, information and behavioral data can range from individual system processes on a single computer system to networked components in an entire branch of infrastructure. Figure 1 provides an overview of the interplay of the three main layers – service, information, and data – and sketches the link to the APT RPG core model. Below axioms describe the cornerstones of our model. More in-depth definitions of terms can be found in Section 3.3.

- **Axiom 1** – *Each service maintains and is maintained by information.* Specifically, a *service* ($\langle Victim \rangle$ class, see Section 3.3) is controlled by *configuration* (technical or organizational settings and policies controlling the service's functions and parameters) while it maintains *knowledge* that either represents potentially sensitive information relevant to the service owner, or the configuration itself. Information is protected by *controls* (technical or organizational measures, see $\langle Enabler \rangle$ class) that focus on safeguarding *confidentiality* and *integrity*.
- **Axiom 2** – *Services are both consumers and providers for other services.* A service ($\langle Victim \rangle$) is an interdependent entity that consumes and/or provides functionality for other entities. The *status* (ability to provide functionality to a consumer) of a service and its place in a hierarchical tree of services is protected by controls mitigating *availability* attacks. Erroneous behavior of a provider negatively impacts its consumers.
- **Axiom 3** – *Service behavior is described by events.* Both nominal (baseline) and anomalous (attacker) behavior is synonymous to one or several *events* ($\langle Event \rangle$ class) captured and pre-evaluated by a dedicated *detection measure*. Events are the link between the attacker/defender model, the APT RPG core system, and actual monitoring data. In the context of the game, a sequence of these behavioral snippets are referred to as *actions X*.

Based on these premises, we can tell straightforward *attack stories* with significant depth and level of detail, which we later build upon.

For example, an operating system process might trigger the event $Evt(operation, layer, argument)$ = `create-data-dropped.exe`, followed by the event

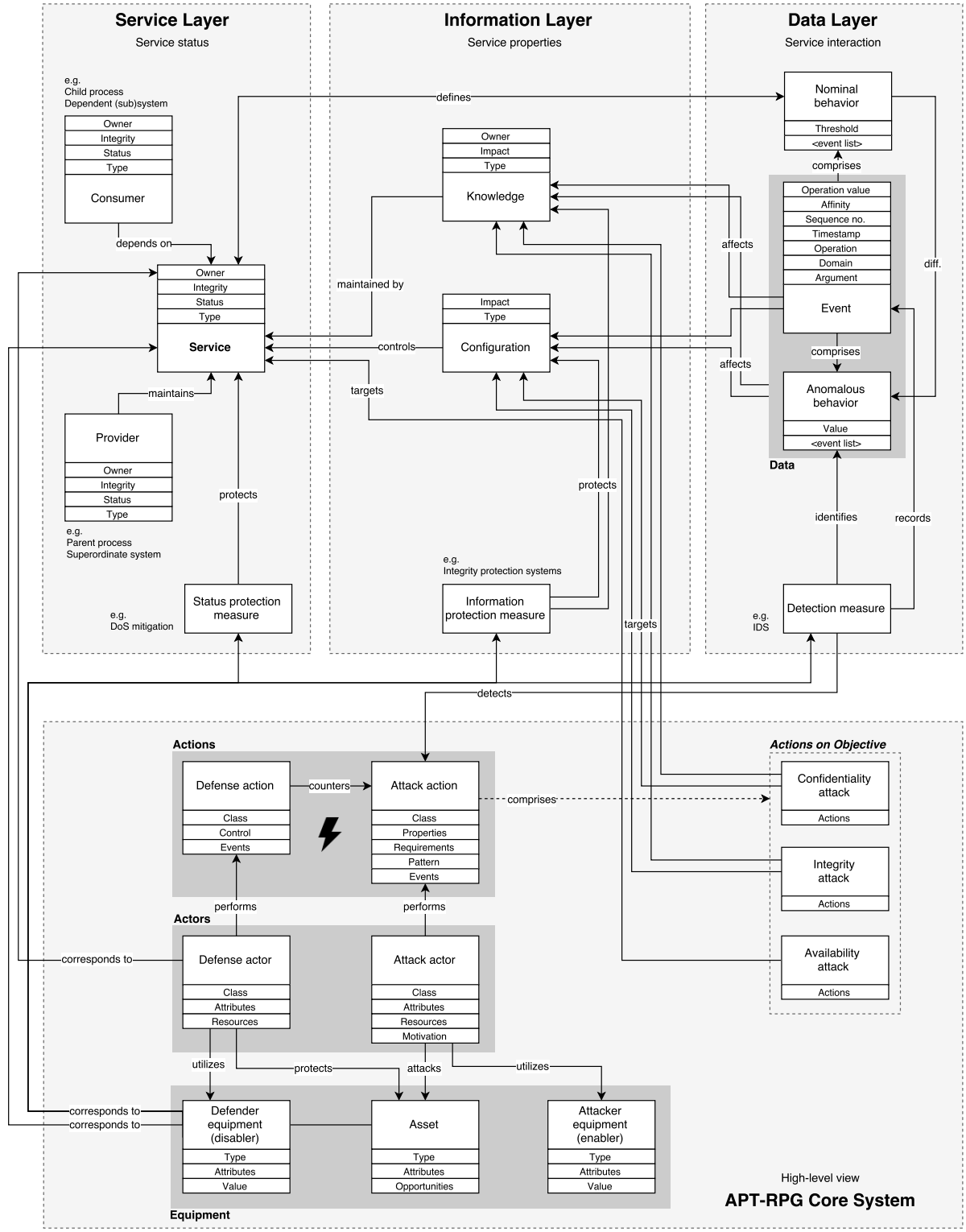

Figure 1: High-level view on the APT-RPG meta model. The top half represents the layers of the base attacker/defender model while the lower portion depicts the APT RPG gamified rule system.

`start-operation-dropped.exe`, which results in an anomaly with a specific value denoting its deviation from a set baseline. Here, this occurrence would affect the confidentiality of user-specific knowledge that is handled by the modeled service `chrome.exe`, thereby changing integrity of the service and its children to 'compromised', while its status ('running') remains unaffected.

On a less granular IT service level, a generic event `destroy-config-webserver` caused by the purposeful alteration of a web server's settings would affect the configuration on an availability level, which in turn affects the electronic campus service of an education organization, changing its status and the status of its children to 'stopped'/'compromised'.

Since we want to not only sketch attack scenarios but also bridge the gap between actors, assets, vulnerabilities, concrete attack patterns, events and countermeasures, we extended the model with formally defined *actions*, which in turn shape the foundation for the APT RPG game model. In the following, we detail some of the game theoretic principles of our model before defining classes and mappings of model components.

3.2 Game Principles

Instead of relying solely on decision models, adversarial behavior can be studied using game situations (Lewis, 2014). Techniques associated with game theory, which is defined as "the study of models of conflict and cooperation between intelligent, rational decision-makers" (Roger, 1991), can be used to model multi-player scenarios that allow participants to pursue goals and rewards by acting in the most favorable, risk-adverse, or cost-effective way possible. The outcome of a game presents the actors with a strategy for resource allocation, risk minimization, or a general strategy to meet a certain attack or defense objective.

Even though it is a model/game hybrid, our RPG can be categorized according to several principles of game theory, as summarized by Roy et al. (2010). Specifically, the APT RPG has the following properties:

- **Non-cooperative nonzero-sum game**: Opposition between players is an integral part of the current design: Player 1 (attacker) is always combats player 2 (defender) and tries to achieve adverse goals by stealing information, manipulating the integrity of data or systems, or by shutting them down entirely. Even though actions are not typically assigned points that are symmetrically gained/lost (making it nonzero-sum), it can be argued that the mechanism of asset compromise is in fact a zero-sum game, where the defender loses integrity while the attacker gains an advantage. In other situations, win/loss is represented by an increase or decrease of attributes or action success and detection chance. These bonuses are one-sided, yet always shift the balance between the players away from equilibrium.
- **Asymmetric strategy**: The strategy sets of the two players are not identical – the attacker draws from a different pool of actions than the defender. This stems from the difference in goal and purpose: Attackers will attempt to penetrate a system using malware and by exploiting vulnerabilities, while a defender tries to counter these actions by implementing technical and organizational controls.
- **Dynamic/extensive game with static elements** (Roy et al., 2010): While the game uses sequential moves characteristic for dynamic games, the second player typically remains unaware of the first player's actions, making the model bear some resemblance to a strategic setting where players act simultaneously and in secret. At its core, the APT RPG remains dynamic – emphasized by its multi-stage nature.
- **Imperfect incomplete information**: As stated above, player 1 does not necessarily know the moves previously made by the attacker. It is in fact vital to his or the success of player 2 that actions performed remain secret, thereby potentially causing the defender to make imperfect decisions. At the same time, the general set of strategies is known to both sides, however the exact payoff in a certain situation is not, due to the lack of information about past activity and their impact on success and detection chances (incomplete information).
- **Bayesian formulation of static elements**: In the RPG, players have incomplete information on the other players, especially when it comes to actions and strategies, which are derived from the attacker's type and ultimate objective. There is, however, a fixed probability that players, being one of 8 available classes, need to conduct/ defend against one of three kinds of attacks on a finite set of assets in order to win the game.
- **Finite & discrete**: While some action combinations are continuous in nature, the general action/reaction game follows a discontinuous sequence. The number of game turns is limited by an exhaustible resource – Initiative (i.e. time efficiency).

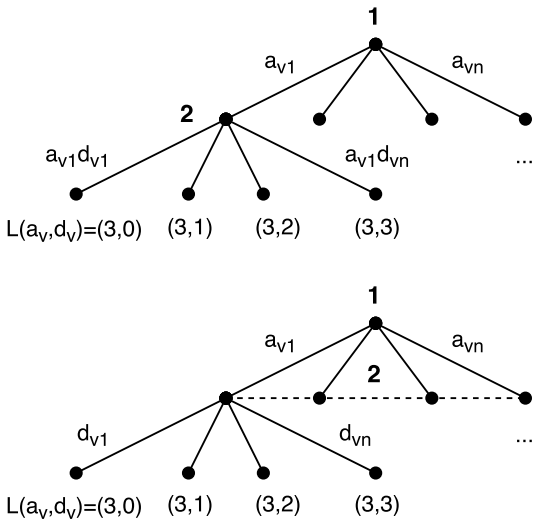

Figure 2: Information set of the defending player for victim system compromise through attacker action a_{v1}, shown in extensive form (Kuhn, 2009). In the above case, player 2 successfully unveils a_{v1}, giving him the chance to specifically counteract the gain $L(a_v)$ of player 1 by increasing his own score $L(d_v)$ through defense action d_{v1}. Below tree represents the limited information set for player 2, when a_{v1} remains undetected.

Following Roy et al.'s taxonomy, our game can be classified as non-cooperative, dynamic game with imperfect and incomplete information that draws from static elements of Bayesian formulation. Akin to You and Shiyong (2003), our two-party hybrid zero-sum game G is defined by the triplet:

$G = (A, D, F)$, where:

- A is a set of strategies of player 1 (attacker),
- D is a set of player 2 (defender) strategies, and
- L is, for select game concepts (see below), a real-valued function $A \times D$. Therefore, $L(a,d)$ is a real number for every $a \in A$ and $d \in D$.

During the game, player 1 chooses action $a \in A$ targeted at victim $v \in V$, which is kept secret unless player 2 meets detection requirements (see Figure 2). If action a_v is conducted successfully, player 1 gains a number of points $L(a_v)$ representing the level of victim compromise, which are directly or indirectly deducted from the respective defender's tally (payoff). Independently from the outcome of the detection attempt, player 2 chooses $d \in D$ for victim $v \in V$, attempting to counter the (assumed) action a_v. Points $L(d_v)$ are computed, fully negating $L(a_v)$ in the best of cases. Specifically, this principle applies to victim system integrity and status (see Figure 2 as well as the $\langle Victim \rangle$ definition in Section 3.3), success chance of hostile attacks and defensive actions (see $\langle Properties \rangle$ and $\langle Enablers \rangle$/$\langle Disablers \rangle$ below), and some other attribute or resource bonuses or penalties.

It is important to bear in mind that the RPG's victory conditions are only indirectly affected by these shifts and that an increased numeric distance from the equilibrium of factors other than victim system integrity and status does not always guarantee one player's domination over the other. In fact, victory is determined by the exhaustion of available (temporal) resources or the successful compromise of the victim asset within an allotted time window.

In the following, we introduce the class specifics of our model and define the concept of an action.

3.3 Definitions

Actions are at the core of the joint model and link real-world service and actor behavior to concrete data points such as observable attack patterns or event sequences. Formally, an action X is defined as n-tuple of typical length $n = 11$, whereas the model's flexibility allows for the omission of unneeded elements. Simply put, an action is performed by an actor and further enabled or disabled by equipment, policies, and tools. It is assigned a category within the model, including usage requirements, properties pertaining to its success, and a detection chance. Attack patterns and associated events tie it to measurable real-world data points.

$$
\begin{aligned}
X = \langle & \\
& \langle AttackActor\langle Class, Motivation, Attributes, \\
& Resources\rangle\rangle, \\
& \langle DefenseActor\langle Class, Attributes, Resources\rangle\rangle, \\
& \langle Enabler\langle Type, Effect, Attributes, Name\rangle\rangle, \\
& \langle Disabler\langle Type, Effect, Attributes, Name\rangle\rangle, \\
& \langle Victim\langle Type, Name, Exposure, Parent, \\
& Configuration, Knowledge, Status, Integrity\rangle\rangle \\
& \langle AttackClass\langle Stage, PatternClass, Mode\rangle\rangle, \\
& \langle DefenseClass\langle Category, ControlClass, \\
& ActionClass\rangle\rangle, \\
& \langle Requirements\langle *Actor\langle Attributes, \\
& Resources\rangle, Victim\langle Exposure, Integrity\rangle\rangle, \\
& \langle Properties\langle Sophistication, SuccessChance, \\
& DetectionChance\rangle\rangle, \\
& \langle AttackPattern\langle Impact, ID\rangle\rangle, \\
& \langle Events\langle Type, Time, Sequence, Parent, \\
& Operation, Argument\rangle\rangle \\
& \rangle
\end{aligned}
$$

AttackActor and *DefenseActor* correspond to the actor types introduced in the `ThreatActorType` vocabulary schema of the STIX threat information language (Barnum, 2012). Each attacker is described as a unique class with their own motivation, primary attributes in the form of Sophistication (measure of an actor's skill, *SO*), Determination (measure of actor motivation and the strength of their cause, *DE*) and Wealth (financial resources of an actor, *WE*), as well as operational resources such as Initiative (*INI*) and Insight (*INS*): Initiative (i.e. time efficiency) is an attribute derived from *SO* and *DE* that determines the number of overall actions each actor can perform. Each action in the strategy set has a time requirement, making this attribute the main resource for all attack and defense activities. Insight, on the other hand, measures the knowledge gained about their opponent and thereby increases the overall chance of success when attacking/defending.

The currently modeled $\langle AttackActor \rangle$ classes in the RPG include: Cyber Espionage Operations (*TH*), Hacker (white (*EX*), gray (*RO*), black hat (*RA*)), Hacktivist (*CR*), State Actor/Agency (*OP*), Insider (*IN*), and Disgruntled Customer (*PR*).

Each attacker class has a range of possible motivations. These goals represent the overall objective of the attack and provide additional context for determining attacker attributes and skills. They correspond to the STIX Threat Actor `Motivation` vocabulary and encompass various ideological goals ($id.*$), as well as ego-centered (*eg*), financial (*fi*), military (*mi*), opportunistic (*op*), and political (*po*) motivations. The actor creation routine of the RPG supports automation of the entire process and provides percentages defining likely actor/goal combinations.

$$
\begin{aligned}
&AttackActor = \langle \\
&\quad \langle Class\{TH, EX, RO, RA, CR, OP, IN, PR\}\rangle, \\
&\quad \langle Motivation\{id.*, eg, fi, mi, op, po\}\rangle, \\
&\quad \langle Attributes\langle SO\{1..n\}, DE\{1..n\}, WE\{1..n\}\rangle\rangle, \\
&\quad \langle Resources\langle INI\{1..n\}, INS\{1..n\}, \\
&\quad \langle Enabler\rangle\rangle\rangle \\
&\quad \rangle
\end{aligned}
$$

Following the same approach, $\langle DefenseActor \rangle$ classes include the three primary sectors (*CP*, *CM*, *CS*), infrastructure (*IF*), military (*MI*), state actors/agencies (*SA*), the education sector (*ED*), and private individuals (*PI*).

$$
\begin{aligned}
&DefenseActor = \langle \\
&\quad \langle Class\{CP, CM, CS, IF, MI, SA, ED, PI\}\rangle, \\
&\quad \langle Attributes\langle SO\{1..n\}, DE\{1..n\}, WE\{1..n\}\rangle, \\
&\quad \langle Resources\langle INI\{1..n\}, INS\{1..n\}, \\
&\quad \langle Disabler\rangle\rangle\rangle \\
&\quad \rangle
\end{aligned}
$$

So-called *Enablers* represent attack tools and vulnerabilities employed by the aggressor. They have specific effects on the target system and come with a wide range of properties (e.g. level of impact or maturity of a vulnerability-type Enabler (see CVE/CVSS mapping in Section 4)) and prerequisites (e.g. privileges and level of user interaction required) that need to be considered. Attacker equipment ($ATT.*$) subsumes mostly attack tools (*MPT*, *Pwd*), OS, application and (wireless) network scanners (*Sca*), various types of malware (*Mal*), and vulnerabilities (*VUL*) that modify the chance of success against specific target assets or actors.

Some enablers and disablers only target or apply to specific equipment categories. The general $\langle Effect \rangle$ of using equipment (or attack actions for that matter) is described by the respective subclass. Typically, various resources, attributes, and detection/success ($*DC/*SC$) probabilities are either increased ($inc*$) or decreased ($dec*$) upon use, which further impacts future events and the overall course of the game. For the $\langle EffectTarget \rangle$, we generally differentiate host-based (*H*), network-based (*N*), industrial (*I*), mobile (*M*), and third-party (*T*) equipment. Enablers have an adverse $\langle Impact \rangle$ on the CIA status of the victim (more details below).

$$
\begin{aligned}
&Enabler = \langle, \\
&\quad \langle Type\{ATT.*, VUL.*\}\rangle, \\
&\quad \langle Effect\langle Type\{incSC, decDC\}, EffectValue\{1..n\}, \\
&\quad EffectTarget\{H, N, I, M, T\}\rangle\rangle, \\
&\quad \langle Attributes\langle Sophistication\{1..n\}, Level\{1, 2\}\rangle, \\
&\quad \langle Properties\langle Privileges\{high, low\}, \\
&\quad UserInteraction\{none, required\}, \\
&\quad \langle Impact\langle C\{low, med, high\}, \\
&\quad I\{low, med, high\}, A\{low, med, high\}\rangle, \\
&\quad Maturity\{unproven, PoC, functional, high\}\rangle\rangle\rangle, \\
&\quad \langle Name\{\_ToolName\_\}\rangle \\
&\quad \rangle
\end{aligned}
$$

Defenders use *Disablers* to thwart their adversary. Such defender equipment encompasses assets to be protected (*AST*), security policies (*POL*), fixes

(*FIX*) that counter vulnerabilities, as well as tools that increase security (*SEC*.∗, e.g. through prevention (*Pre*)), detect (*Det*), delay (*Del*), or generally hinder the attacker (*Cnt*). The respective effects aim to reverse or mitigate the damage caused by attacker equipment and actions.

$$
\begin{aligned}
&Disabler = \langle, \\
&\quad \langle Type\{AST.*, SEC.*, POL.*, FIX.*\}\rangle, \\
&\quad \langle Effect\langle Type\{decSC, decSC\}, EffectValue\{1..n\}, \\
&\quad EffectTarget\{H, N, I, M, T\}\rangle\rangle, \\
&\quad \langle Attributes\langle Sophistication\{1..n\}, Level\{1,2\}\rangle\rangle, \\
&\quad \langle Properties\langle Maturity\{official, temporary, \\
&\quad workaround\}\rangle\rangle, \\
&\quad \langle Name\{\_SystemName\_\}\rangle \\
&\quad \rangle
\end{aligned}
$$

The *Victim* class describes the system that might be targeted by the attacker, may that be a specific asset or a disabler. Furthermore, victim information differentiates between internal and exposed systems, which can be accessed directly from within the DMZ, and identifies service status, integrity, and systemic parents that represent consumers and providers.

In each gamified scenario, the attacker attempts to achieve his or her goal by compromising a victim in a specific fashion, i.e. one tries to compromise one of the defender's assets (change its $\langle Victim\langle Integrity\rangle\rangle$ to 'compromised' or its $\langle Victim\langle Status\rangle\rangle$) to 'stopped'). There are three kinds of possible attacks on any asset, in accordance to the CIA triangle (Whitman and Mattord, 2011): Confidentiality attacks (theft of information), integrity attacks (altering of information), and availability attacks (system status changes), which are part of the $\langle AttackClass\rangle$ component below.

$$
\begin{aligned}
&Victim = \langle \\
&\quad \langle Type\{Asset, SecuritySolution\}\rangle, \\
&\quad \langle Name\{\_SystemName\_\}\rangle, \\
&\quad \langle Exposure\{internal, exposed\}\rangle, \\
&\quad \langle Parent\{\langle Victim\rangle, \langle Disabler\rangle\}\rangle, \\
&\quad \langle Configuration\{tech, org\}\rangle, \\
&\quad \langle Knowledge\{Information, \langle Configuration\rangle\}\rangle, \\
&\quad \langle Status\langle OperationStatus\{running, stopped\}\rangle, \\
&\quad \langle Integrity\{nominal, affected, compromised\}\rangle\rangle \\
&\quad \rangle
\end{aligned}
$$

AttackClass contains categorization into the APT kill chain (sub-)stage ($\langle Stage\rangle$), as well as the class of attack patterns as defined by the CAPEC link categorization introduced below. $\langle Mode\rangle$ identifies the optional CIA goal and impact rating of the action. Respectively, the *DefenseClass* refers to the NIST link categories which allows the assignment of a mitigating control to the modeled attack. See Section 4 for more information about categorization.

$$
\begin{aligned}
&AttackClass = \langle \\
&\quad \langle Stage\{R.*, W.*, D.*, E.*, I.*, C.*, A.*\}\rangle, \\
&\quad \langle PatternClass\{IG, IN, SE, SA, FA, BF, IA, DM, \\
&\quad PR, CO\}\rangle, \\
&\quad \langle Mode\langle Goal\{C, I, A\}, Rating\{low, med, high\}\rangle\rangle \\
&\quad \rangle
\end{aligned}
$$

$$
\begin{aligned}
&DefenseClass = \langle \\
&\quad \langle Category\{organization, information\_system\}\rangle \\
&\quad \langle ControlClass\{IL, CP, AW, SP, FI, AP, AC, DI, \\
&\quad SI, CS\}\rangle \\
&\quad \langle ActionClass\{ACM, ACE, IFE, LEP, REA, \\
&\quad AMO, RST, COM, CCC, COS, COP, COP, AUM \\
&\quad INH, NOM, MES, PAC, SEP, SCP, BOP, CRP, \\
&\quad FLR, MCP, ISM\}\rangle \\
&\quad \rangle
\end{aligned}
$$

Requirements identify the minimum actor attributes and resources needed to conduct the attack, as well as other prerequisites in the form of system exposure. Requirements are optional and depend on the complexity of the respective attack, which is defined through its *Properties*, namely minimum skill (Sophistication *SO*) and the chance of success (*SC*) and detection (*DC*). This information is largely gleaned from CAPEC and CVE.

$$
\begin{aligned}
&Requirements = \langle \\
&\quad \langle *Actor\langle Attr.\langle SO\{1..n\}, DE\{0..n\}, WE\{0..n\}\rangle\rangle\rangle, \\
&\quad \langle *Actor\langle Resources\langle INI\{1..n\}, INS\{1..n\}\rangle\rangle\rangle, \\
&\quad \langle Victim\langle Exposure\{internal, exposed\}\rangle\rangle, \\
&\quad \langle Victim\langle Integrity\{nominal, compromised\}\rangle\rangle \\
&\quad \rangle
\end{aligned}
$$

$$
\begin{aligned}
&Properties = \langle \\
&\quad \langle SO\{1..n\}\rangle, \langle SC\{1..100\}\rangle, \langle DC\{1..100\}\rangle \\
&\quad \rangle
\end{aligned}
$$

Each action corresponds to an *AttackPattern*, which is identified by its ID and its impact on the CIA triangle. Attack patterns link the modeled action to specific hostile activity as described in the CAPEC

schema. Ultimately, each attack pattern is mapped to a set of monitored *Events* with specific underlying operations and arguments, triggers (parents), timestamps, and sequence numbers. The 'pattern' type describes event sequences that directly represent the action, while anomalies describe the behavioral deviation from a baseline. The modeling of unique events closes the gap to the data layer and allows us to lower the level of abstraction to the actual systemic representation. Refer to Section 4 for a mapping example.

$$
\begin{aligned}
&AttackPattern = \langle \\
&\quad \langle Impact \langle C\{low, med, high\}, \\
&\quad I\{low, med, high\}, A\{low, med, high\}\rangle\rangle, \\
&\quad \langle ID\{n\}\rangle \\
&\quad \rangle
\end{aligned}
$$

$$
\begin{aligned}
&Event = \langle \\
&\quad \langle Type\{Pattern, Anomaly\}\rangle \\
&\quad \langle Time\langle Start\{\_timestmp\_\}, End\{\_timestmp\_\}\rangle\rangle, \\
&\quad \langle Sequence\{n\}\rangle, \\
&\quad \langle Parent\{\_ParentName\_\}\rangle, \\
&\quad \langle Operation\{\_OperationName\_\}\rangle, \\
&\quad \langle Argument\{\_OperationArgument\_\}\rangle \\
&\quad \rangle
\end{aligned}
$$

4 Data Mapping

In this section, we specify the mapping of external data to the various classes and categories that are part of our model. This includes the formal link between actions and events, the mapping of APT kill chain to CAPEC attack patterns, the CAPEC to CVSS mapping, and the association of controls to defense actions. Using below information, it is possible to easily extend or adapt the game system to new or updated scenarios in cyber-security and beyond.

4.1 Actions to Events

Each action available in our APT RPG rule system can be modeled using the structure introduced in Section 3. The possible link between in-game actions and real-world events is an integral part of the model. In the following, we exemplify this mapping using the star graph based anomaly detection approach by Luh et al. (2017). In that paper, the authors describe a system for detecting and interpreting abnormal Windows OS behavior expressed through sequences of kernel events defined as $G = (U, V, E)$, where U and V are nodes (parametrized event type) and E is the respective edge (type of operation). Specific attacks (here, the example is limited to APT kill chain phases) can be learned by monitoring malware activity and comparing it to a pre-established baseline of known process behavior. If we apply this principle to CAPEC attack patterns, the accuracy of the interpretation mechanism could be further improved.

In Luh et al.'s system, such a sequence of kernel events could look like this:

Start node (U)	End node (V)	Edge (E)
process-shell.exe	process-drop.exe	start (3)
process-drop.exe	image-library.dll	load (1.5)
process-drop.exe	registry-HKLM/ Software/.../Run	open (0.25)
process-drop.exe	registry-REG_SZ evil.exe	add (0.75)

Table 1: Example event sequence describing the process of making a dropped executable persist through a system restart (CAPEC-564: "Run Software at Logon"). Time stamps and full paths omitted for legibility.

This real-world data could be collected using any kernel monitor or API monitoring tool. More importantly, our APT RPG can seamlessly integrate this information into a gamified scenario. Specifically, our system would model the above event sequence as action, where the $\langle Event \rangle$ class looks like this:

$$
\begin{aligned}
&Event = \langle \\
&\quad \langle Type = Pattern\rangle \\
&\quad \langle Time\langle Start = 16.53.661, End = 16.53.729\rangle\rangle, \\
&\quad \langle Sequence = 1\rangle, \\
&\quad \langle Parent = shell.exe\rangle, \\
&\quad \langle Operation = process\_start\rangle, \\
&\quad \langle Argument = drop.exe\rangle \\
&\quad \langle Sequence = 2\rangle, \\
&\quad \langle Parent = drop.exe\rangle, \\
&\quad \langle Operation = image\_load\rangle, \\
&\quad \langle Argument = library.dll\rangle \\
&\quad ... \\
&\quad \rangle
\end{aligned}
$$

This simple interface makes it easy for analysis to add their own data to the game model. Thanks to its flexibility, the level of abstraction can be freely specified – ranging from the discussed kernel events to high-level behavior such as 'set fire to house'.

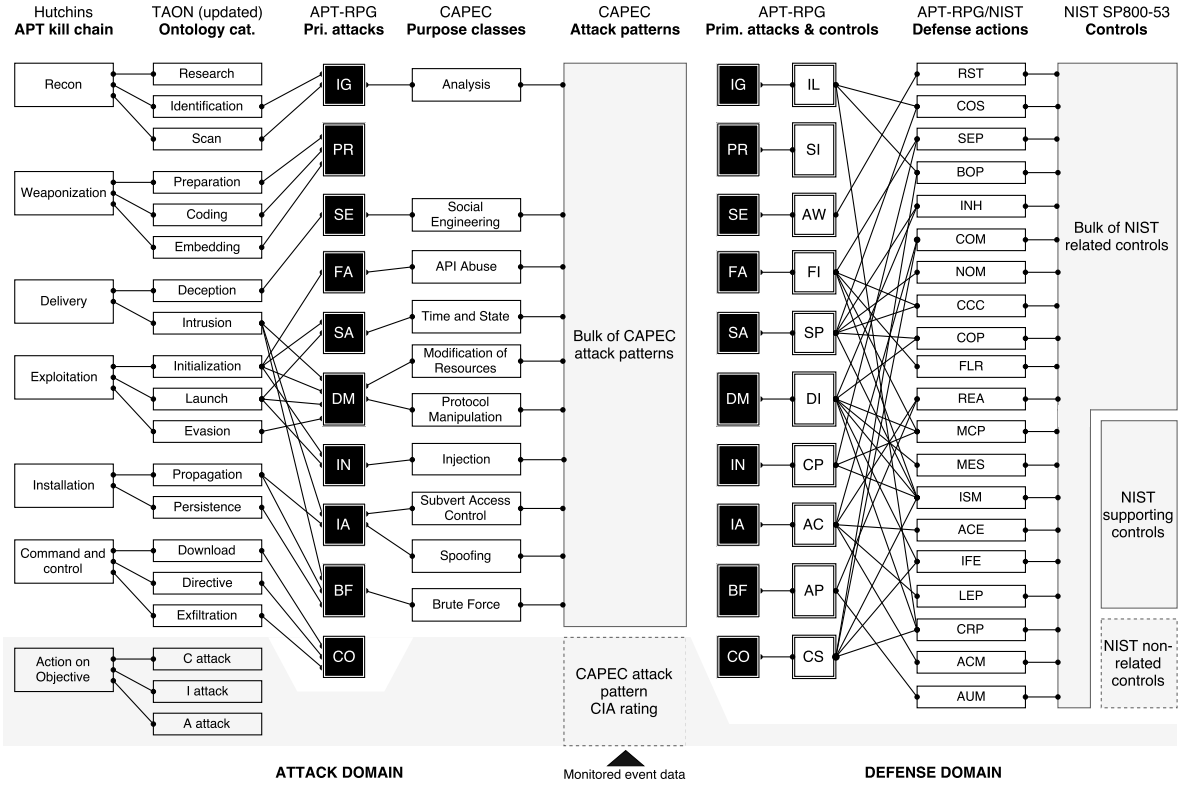

Figure 3: Mapping of NIST controls to CAPEC attack patterns via extended APT kill chain. The introduced link categories based on CAPEC are highlighted in black. A full key and table for all abbreviations/mappings is available on request.

4.2 Kill Chain to Attack Patterns

We use Hutchins et al.'s cyber kill chain 2011 as foundation for the high-level view on our model. For further granularity, we expanded the 7 stages to a total of 19 subcategories that largely adhere to the APT ontology design by Luh et al. (2016). To link kill chain elements to CAPEC attack patterns, we introduced the concept of *primary attacks*, which were abstracted from CAPEC's 'purpose classes' and assigned to kill chain categories. Figure 3 depicts the mapping.

Since the list of attack patterns in CAPEC is partially incomplete or offers too little information in terms of abstraction required for a meta-model, we only considered patterns of 'standard' and 'meta' abstraction level – the 'detailed' class was omitted in this initial iteration of the game. In addition, we opted to remove deprecated attacks and focus only on stable and draft patterns found in version 2.11 of CAPEC[1]. Incomplete patterns with no purpose classes were disregarded as well. Overall, this selection retained a total of 65 representative patterns out of 516. The remainder can easily be added at a later point.

4.3 Attack Patterns to Vulnerabilities

Vulnerabilities are a key component of any intrusion scenario. In order to automatically map CAPEC attack patterns to CVSS vulnerabilities, we took a closer look at these and other related MITRE information exchange standards. Ultimately, we use the 'related weaknesses' information provided by CAPEC to map each pattern to specific weaknesses represented by the Common Weakness Enumeration (CWE) list. CWE "provides a common language for describing security weaknesses in architecture, design, or code"[2]. For example, CAPEC ID 1 ("Accessing Functionality Not Properly Constrained by ACLs") is related to CWE ID 276, 285, 434, etc.).

To close the gap between weaknesses and the more concrete vulnerabilities, we use open source information to map CWE entries to their Common Vulnerabilities and Exposures (CVE) counterpart. For example, CWE-276 ("Incorrect Default Permissions") contains the vulnerabilities CVE-2005-1941, CVE-2002-1713, CVE-2001-1550, CVE-2002-1711, CVE-2002-1844, CVE-2001-0497, and CVE-1999-

[1] https://capec.mitre.org

[2] https://cwe.mitre.org/about/index.html

0426. Armed with this ID, the specific vulnerability can be looked up in the National Vulnerability Database (NVD[3]), where a specific CVSS score is assigned. This multipartite score is the basis for the requirements and impact calculations used in our model (see $\langle Enabler \rangle$ in Section 3). In our example, the score for CVE-2005-1941 can be easily retrieved[4].

4.4 Primary Controls to Defense Actions

The mapping of controls to defense actions and primary controls was automatically computed from the NIST SP800-53r4 standard. To achieve this, we extracted all links of relationship between the individual controls and modeled them as a graph. We subsequently separated each control into one of three categories by their level of degree. Controls with a degree $d >= 20$ were defined as parent *defense actions* (see Figure 3). Controls with degree $d < 20$ and $d >= 1$ form the bulk of related controls which operate on information system and/or organization level and translate to the remainder of the defense strategy set for the respective parent action. The model can be further extended by adding control enhancements that, in turn, are associated with numerous parent and related controls. Figure 4 depicts the relationship between all 224 NIST controls. General measures (NIST suffix *-1) were converted into policies, which are part of the $\langle Disablers \rangle$ class.

4.5 Limitations

There are a number of limitations that need to be considered before employing the described version of the APT RPG for attack analysis, risk assessment, or simply as an awareness game. Most of these factors also offer opportunities for future research.

Firstly, one needs to keep in mind that using external data sources or threat intelligence can be limiting: Attack patterns may outdate due to a lack of database maintenance or novel threats might not be considered immediately after disclosure. For CAPEC specifically, there are patterns that lack CIA impact or purpose information, e.g. CAPEC-2: "Inducing account lockout". This might make it necessary for the player to expand the repository of attack strategies/actions with their own information.

Secondly, the assignment of primary controls to defense actions (see Figure 3) is currently done semi-manually by a group if IT security experts. Automated mechanisms have proven to be too inaccurate during initial experiments that utilized a natural language approach using several IT security glossaries. Because of this limitation, the attack–defense mapping of data sources that are significantly larger than the NIST standard might not be feasible.

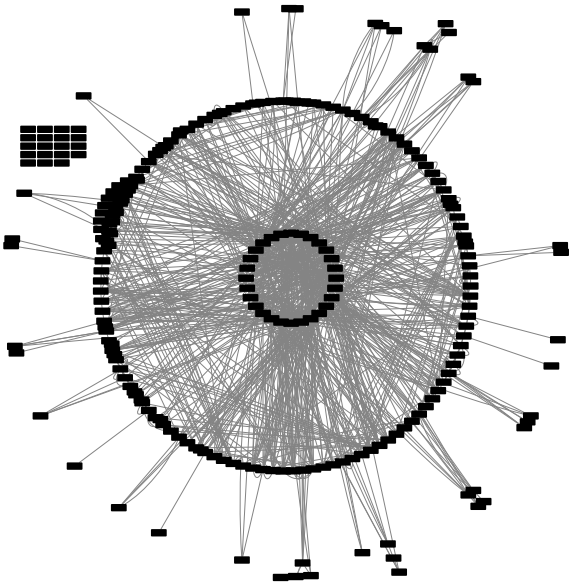

Figure 4: Relations between NIST controls rendered in Cytoscape (Shannon et al., 2003). The inner circle represents controls with a degree of ≥ 20. Elements beyond the outer circle have a degree of 1, while the isolated nodes to the left are not linked to any other controls (orphan controls).

Another minor drawback of the NIST control approach is the small number (19) of orphan controls, which are not related to any other countermeasures. Currently, these controls are not considered in the game, since they would have to be manually added and assigned a specific category, levering out most mechanisms of automation currently in place.

On the CAPEC side, we do not currently restrict attacks to a specific target type like we do for equipment, since CAPEC does not offer a clean way to assign systems (hosts, network, industrial components, etc.) to a particular pattern. That means that e.g. the *BF* primary attack such as CAPEC-49: "Password Brute Forcing" can, in the game, be used on an arbitrary victim without further distinction into target categories. This limitation might in some cases simplify a hostile action at the expense of realism. Countering this drawback is possible, but currently requires manual intervention: Depending on the description of the respective attack pattern, the information provided in CAPEC's database (namely the 'summary' and 'example instances' columns) can be parsed and assigned one of the equipment type categories used for assets ($\langle EffectTarget \rangle$).

[3] https://nvd.nist.gov

[4] https://nvd.nist.gov/vuln/detail/CVE-2005-1941

5 Conclusion

We have presented the design of a gamified meta model that can be used to train personnel, assess risk mitigation strategies, and compute new attacker/defender scenarios in arbitrary IT infrastructures. While the model itself is extremely flexible and supports varying levels of granularity, the initial game prototype design based upon this model utilizes accepted taxonomies and security standards to support out-of-the-box organization-level gameplay for simulating cyber-attacks on various types of local or networked assets. Our data mapping mechanisms enable domain experts to easily extend the system with new actors, actions, and (mitigating) equipment. We also exemplified how real-world data such as OS kernel events can be linked to the model.

The development of the first educational game prototype based on the introduced model has been completed. Ultimately, it is planned to evaluate both the physical release candidate as well as a simulation app that will allow us to automatically compute new attack stories and identify systemic weaknesses in only slightly abstracted infrastructures.

Next to simulation, the APT RPG offers a solid foundation for the development of an ontology for targeted attacks, which can be populated by both threat information sources as well as host and network monitoring data. The synergies between the data-centric and model-based game system will significantly aid in understanding and closing the semantic gap.

ACKNOWLEDGEMENTS

The financial support by the Austrian Federal Ministry of Science, Research and Economy and the National Foundation for Research, Technology and Development is gratefully acknowledged.

REFERENCES

Barnum, S. (2012). Standardizing cyber threat intelligence information with the Structured Threat Information eXpression (STIX^{TM}). *MITRE Corporation*, 11:1–22.

Caltagirone, S., Pendergast, A., and Betz, C. (2013). The diamond model of intrusion analysis. Technical report, Center for Cyber Intelligence Analysis and Threat Research, Hanover.

Cook, A., Smith, R., Maglaras, L., and Janicke, H. (2016). Measuring the risk of cyber attack in industrial control systems. BCS eWiC.

Holmgren, A. J., Jenelius, E., and Westin, J. (2007). Evaluating strategies for defending electric power networks against antagonistic attacks. *IEEE Transactions on Power Systems*, 22(1):76–84.

Hutchins, E. M., Cloppert, M. J., and Amin, R. M. (2011). Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Leading Issues in Information Warfare & Security Research*, 1:80.

Kuhn, H. W. (2009). *Lectures on the Theory of Games*. Princeton University Press.

Lewis, T. G. (2014). *Critical infrastructure protection in homeland security: Defending a networked nation*. John Wiley & Sons.

Luh, R., Schrittwieser, S., and Marschalek, S. (2016). TAON: An ontology-based approach to mitigating targeted attacks. In *Proc. of the 18th Int. Conference on Information Integration and Web-based Applications & Services*. ACM.

Luh, R., Schrittwieser, S., Marschalek, S., and Janicke, H. (2017). Design of an anomaly-based threat detection & explication system. In *Proc. of the 3rd Int. Conference on Information Systems Security & Privacy*. SCITEPRESS.

Miura-Ko, R. A., Yolken, B., Bambos, N., and Mitchell, J. (2008). Security investment games of interdependent organizations. In *Communication, Control, and Computing, 2008 46th Annual Allerton Conference on*, pages 252–260. IEEE.

Nguyen, K. C., Alpcan, T., and Basar, T. (2009). Security games with incomplete information. In *Communications, 2009. ICC'09. IEEE International Conference on*, pages 1–6. IEEE.

Roger, B. M. (1991). Game theory: Analysis of conflict.

Roy, S., Ellis, C., Shiva, S., Dasgupta, D., Shandilya, V., and Wu, Q. (2010). A survey of game theory as applied to network security. In *System Sciences (HICSS), 2010 43rd Hawaii International Conference on*, pages 1–10. IEEE.

Shannon, P., Markiel, A., Ozier, O., Baliga, N. S., Wang, J. T., Ramage, D., Amin, N., Schwikowski, B., and Ideker, T. (2003). Cytoscape: A software environment for integrated models of biomolecular interaction networks. *Genome research*, 13(11):2498–2504.

Syed, Z., Padia, A., Finin, T., Mathews, M. L., and Joshi, A. (2016). UCO: A Unified Cybersecurity Ontology.

Undercoffer, J., Pinkston, J., Joshi, A., and Finin, T. (2004). A target-centric ontology for intrusion detection. In *18th International Joint Conference on Artificial Intelligence*, pages 9–15.

Whitman, M. E. and Mattord, H. J. (2011). *Principles of information security*. Cengage Learning.

You, X. Z. and Shiyong, Z. (2003). A kind of network security behavior model based on game theory. In *Parallel and Distributed Computing, Applications and Technologies, 2003. PDCAT'2003. Proceedings of the Fourth International Conference on*, pages 950–954. IEEE.