Modelling Hybrid Cyber Kill Chain

Wen Zeng and Vasileios Germanos

Cyber Technology Institute, School of Computer Science and Informatics
De Montfort University, Leicester LE1 9BH, U.K.
{wen.zeng.wz,vasileios.germanos}@gmail.com

Abstract. The Cyber Kill Chain (CKC) model is an industry-accepted methodology for understanding how an attacker will conduct the activities necessary to cause harm to the organizations. CKC has seen some adoption in the information security community. However, acceptance is not universal, with critics pointing to what they believe are fundamental flaws in the model. Therefore, an effective understanding of the CKC will greatly assist the information security professionals in establishing strong controls and countermeasures that will serve to protect organizations' assets. In this paper, we will develop a new framework – Hybrid Cyber Kill Chain which is based on CKC, and then a formal model will be built to analyse this framework. In this framework, we will investigate the strength level of threat actors and defenders. The framework can be captured by coloured Petri nets. This study can help security professionals find out the fundamental flaws of the basic CKC. Our work also can help business organizations take actions to deal with incidents to eliminate or minimize the impact and establish strong controls and countermeasures.

Keywords: Petri nets · Cyber security · controls · Cyber Kill Chain · Cyber attack

1 Introduction

Cyber Kill Chain (CKC) is developed by Lockheed Martin, which is an american global aerospace, defense, security and advanced technologies company (https://www.lockheedmartin.com) [12]. CKC is an intelligence-driven, threat-focused framework (see Figure 1) for the defence of computer networks. It takes the view point of an adversary, which shows the stages of what must be achieved in order for an attacker to be successful. This could give network defenders the advantage in fighting cyber attackers by establishing strong controls and countermeasures. The cyber kill chain model has seen some adoption in the information security community [14]. However, acceptance is not universal, with critics pointing to what they believe are fundamental flaws in the model [19]. Therefore, it is necessary to build a formal model to analyse the CKC which can help us find out the fundamental flaws.

Recently, there is a significant interest in analysing CKC. In Black Hat 2013, Patrick Reidy presented a variety of methods to combat insider threats at the FBI [21]. These methods are based on CKC framework. It was stated that although

CKC is an excellent framework for the defence of external attacks, it is not so suitable for insider threats. In general, there is a misunderstanding regarding the type of threats in an organisation. It is often the case that a threat is not an external one (e.g., some hacker) but an insider (e.g., people who joined the organisation) [21]. Dealing with insider threats is not a task that can be supported by firewalls and antivirus. This is not a simple cyber security problem and these defence mechanisms do not apply as no rules are being broken. As a result, they proposed a multidisciplinary approach that extends the CKC framework, the Insider Threat CKC.

In [6], the author argued that the scope of today's cyber threats extends far beyond that of CKC. Today's networks are facing various threat vectors, e.g., social engineering; insider threat; and intrusion based on remote access. All these threat vectors are not related to malware or payload. However, CKC is based on perimeter/malware-focused thinking. Moreover, on an attack time scale, the steps of CKC (see Section 3) are disproportionate. The steps from Reconnaissance to Command and Control comprise a small part of an attack, compare to the last two steps ("Command & Control" and "Action on Objectives") which are the biggest two. Consequently, most of the attention must be paid between these two last steps. The reason behind this is that once the attacker penetrates the perimeter defined by CKC, the prevention solutions, e.g., firewalls and antivirus, cannot be applied. Thus, one should focus on deploying a breach detection system that automatically detects and analyses any change in user/computer behaviour that indicates a breach.

In [15], the author presented three key benefits of applying the CKC model. Firstly, CKC enhances the understanding of the capabilities an organisation has; any gaps that are not covered by tools; and threat actors. Secondly, CKC can be leveraged in post-incintent analysis by break down the attack in a systematic way. Thirdly, a big advantage of CKC model is that it can provide, efficiently, an explanation about how a complex situation evolved, using use-case diagrams. However, there is some criticism about CKC ability to handle an insider breach [21]. In this case, CKC framework can be effectively used as an insider, who would conduct the same steps of CKC having as a final goal to achieve the "Actions on Objectives" stage. The US Senate Committee on Commerce, Science, and Transportation published a report [28] regarding the analysis of a cyber attack against Target, one of the largest retail companies in the US. In this case, the attacker managed to gain access to Target's network using as an insider a third-party vendor of the company. The analysis of this report was conducting using the principles of CKC. All in all, CKC is a powerful framework that can resolve highly complex attack scenarios and be customised/extended, accordingly, to fit an organisation's specific requirements.

In [11], Laliberte proposed a version of CKC aiming to model types of attacks that take advantage of malicious websites to target outdated software on its visitors systems. This version proposed an additional stage the Lateral Movement, which sits between Command & Control and Actions on Objectives. This new stage expresses the move of the attacker to bigger targets using the target network after some of its systems has been compromised. In addition, the author states that one cannot provide defence mechanisms against the Weaponisation stage, so this stage is redundant and is not considered in the proposed kill chain.

In [20], Nachreiner proposed a kill chain where Weaponisation stage is not considered as well for the same reason as in [11]. The Installation stage is replaced by Infection with the intuition that attackers can use alternative methods apart from installation. To capture the way a malicious code can spread in the target's network by leveraging its initial network access, it is proposed to embed the notions of Lateral Movement and Pivoting within the Actions on Objectives stage. The term pivoting describes the exploitation of a system to connect other external or internal systems, in which direct access is not feasible.

MITRE, a non-profit research centre, proposed a more structured framework that can improve the analysis of cyber attacks, the Adversarial Tactics, Techniques & Common Knowledge for Enterprise (ATT&CK) [17]. In a nutshell, this framework consists of stages in a non-sequential style to provide more detailed information regarding the actions an attacker performs once inside the target's network. Among other stages, ATT&CK introduces the Persistence stage, which denotes the attacker's persistence to accomplish an action.

The aim of this paper is to develop a framework to analyse how attackers conduct the activities to organizations based on CKC, and then the proposed framework will be formally modelled and analysed. The strength level will be introduced to attackers and defenders in the model, and then we will demonstrate how Petri net techniques can be used to model the proposed framework.

This paper is organized as follows. Section 2 is related work. Section 3 provides the basic definitions used throughout this paper. In Section 4, we will introduce a new framework for CKC - Hybrid Cyber Kill Chain (HCKC), and a formal model will be developed for HCKC. The basic definitions relating to Petri nets are given in Section 5. Section 6 outline how Petri nets could be used to support analysing the proposed framework. Section 7 concludes the paper.

2 Related Work

Large organizations use a large number of business processes, which in many cases are unique, to achieve their goals and objectives. Moreover, their concurrent structure and complexity is a challenge for system architects and security analysts to develop efficient and secure systems. To alleviate their workload various model-based security analysis techniques have been proposed.

Our study is not the first instance of model-based security analysis of hackers' behaviours. In [23], attack trees have been used to describe how sets of events can constitute a security compromise. Attack trees are useful for thinking multiple ways that a hacker can reach an attack goal. In [24, 25], the authors introduced attack graphs which were extended to attack trees by introducing state to analysis. Although, attack trees enable state-based analysis, they do not take into account the different attack goals and preferences of individual attackers. Attack trees and attack graphs are popular modelling approaches because they are good at describing an attack in an intuitive visual way, show all attack paths within a broad picture, and can lead to useful mathematical analysis if in the nodes are assigned values. However, attack trees and attack graphs are proceeding in sequential steps. They focus on a single goal and a single attacker.

In [16], McDermott proposed that Petri nets are more appropriate for cyber attack modelling than attack trees, as Petri nets do not have the limitations of attack trees and attack graphs, and can capture complex attacks with multiple steps. In addition, Petri nets are better at capturing concurrent actions in the progression of an attack. For instance, multiple instances of attackers and attack goals can be captured and analysed. In [29], the authors proposed that coloured Petri nets are suitable for hierarchical attack modelling, as they more expressive.

There are some existing work aimed at analyzing security properties using Petri nets: In [3, 2], the authors proposed a Petri net modelling technique can be used to analyse opacity which is a security property in distributed systems. In [27], the author proposed an alternative method of information system risk analysis that supports ongoing information system risk management based upon information system security modelling. Petri nets are used to simulate and analyse the complex information security system behaviour. In [30], the authors proposed a formal threat-driven approach which explores explicit behaviours of security threats as the mediator between security goals and application of security feature. The security threats are modelled by Petri nets which can verify correctness of security threats against intended functions and verify absence of security threats from integrated functions and threat mitigation. In [4], the authors modelled cyber-physical attacks on smart grid using Petri nets, in this study, they proposed a hierarchical method to construct large Petri nets from a number of small Petri nets that can be created separately by different domain experts. In [33], the authors proposed a Petri net model to analyse the threats and user behaviour in the business organizations. This study focused on the user behaviour, threats and security policies in a dynamic environment. In [9], the authors proposed an approach to model malware behaviour using coloured Petri nets. In [10, 26], the authors demonstrated how coloured Petri nets can be used to model protocol elements and improve protocol specifications, and how state space exploration can be verify protocol properties. In [34, 32], the authors proposed that Petri nets related techniques can be used to analyse the information flow security in cloud computing systems. In [31], the author proposed a Petri net based methodology to analyse the non-productive time in the organizations by implementing information security technologies.

3 Preliminary Material

In this part, the definition of CKC is presented making it easier to follow the technical content of this paper.

CKC is a framework that helps to understand how a threat actor, such as an attacker, will orchestrate the necessary actions to cause harm to an organisation. There are seven stages in CKC: Reconnaissance, weaponization, delivery, exploitation, installation, command & control, and action on objectives [12, 5].

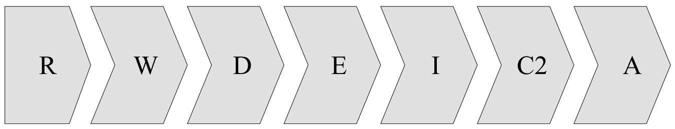

Fig. 1. Cyber Kill Chain - R stands for Reconnaissance, W for Weaponisation, D for Delivery, E for Exploitation, I for Installation, C2 for Command & Control, and A for Action on Objectives.

Stage 1 - *Reconnaissance*: In this stage, the attackers would gather as much information as possible about target organization. For example: names of the employers, their positions, email addresses and IP addresses. They also can purchase information security resources in State and Military level.

Stage 2 - *Weaponisation*: In this stage, the attackers develop exploitation and malware. The attackers might create websites that contain malware, and develop malicious software for a specific platform or purpose (e.g., a vulnerable Adobe Acrobat (PDF) or a Microsoft Office (DOC) file), which designed according to the vulnerabilities discovered during reconnaissance.

Stage 3 - *Delivery*: In this third stage, the attackers transmits the malicious code to the target information system. The attacker might use a spear-phishing attack targeting an internal employee of the organisation, social engineering or some vulnerability of the system to deliver the malware.

Stage 4 - *Exploitation*: The attacker in this phase, utilizing the discovered vulnerabilities, is executing the malicious code on the target network using remote or local mechanisms. The aim is to gain superuser access to the target's information system.

Stage 5 - *Installation*: As soon as the exploitation is successful, the malicious code will install itself onto the target's information system. Then, the malicious code will start downloading additional software when there is available network access. The attacker can maintain continuity of access to the target system by a remote access trojan or creating a backdoor in order to penetrate further into the target network. This practice keeps the delivery payload small making it undetectable.

Stage 6 - *Command & Control*: This stage is also know as C2. Here, the attacker has set up the management and communication code onto the target network. Now, the attacker can fully manage the malicious code and move further into the network, exfiltrate data and conduct destruction or denial of service operations.

Stage 7 - *Action on Objectives*: The actions and objectives of the attacks are dependent on their specific mission (e.g., exfiltrate data and conduct destruction or denial of service operations).

An assimilation of CKC will greatly assist chief information security officers in establishing strong controls and countermeasures that will serve to protect their organization's assets.

4 Hybrid Cyber Kill Chain

In this section, we will introduce a new framework – Hybrid Cyber Kill Chain (HCKC).

In general, the purpose of CKC is to enhance the resilience of an organisation against cyber attacks. Although CKC has been widely adopted by organisations since it has been published, it has been criticized for its weakness in handling insider threats, because it is too focused on the perimeter and malware prevention [21]. This statement is supported also by case studies, where CKC has been applied. For example, [28] shows that the stages of CKC do not capture all possible attack vectors and are not expressive enough to model internal actions within the target network. Consequently, based on these facts, improvements and amendments to CKC have been proposed. Some of these proposed extensions try to address the insider's perspective.

On top of CKC, the proposed kill chain framework aims to provide a holistic approach by taking into account the following factors:

- The persistence and the lateral movement of the attacker;
- The defence actions of the defender;
- The strength level both of the attacker and defender.

The third factor is embedded in each stage of HCKC after the weaponisation. That is because the two first stages are not under the control of the defender. According to this rule, the attacker's strength level must be greater than the one of the defender's in order to proceed to the next stage in the kill chain.

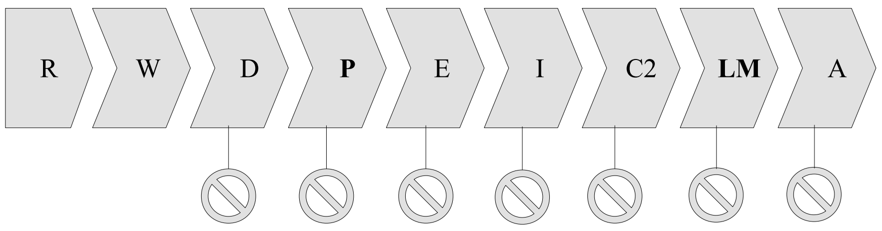

Fig. 2. Hybrid Cyber Kill Chain – P stands for Persistence, LM for Lateral Movement, and cyrcles denote the defender.

HCKC consists of two parts. Initially, we have the CKC augmented with extra two stages – Persistence and Lateral Movement. Then, we map to each attack stage an extra stage that corresponds to the defensive actions of the defender, as depicted in Figure 2. Moreover, a strength level is assigned to each stage.

In practice, HCKC is not only useful for post-incident analysis of attacks. A formal model of the HCKC can provide security metrics that can be utilized by system architects to make trade-off decisions involving system security. For instance, there is a growing need for efficient and timely incident response, as the number of targeted and complex cyber attacks on business organizations is increasing. This need becomes more urgent as the information collecting from intrusion detection systems often must be processed manually to enable decisions on how to respond for thwarting attacks. In many cases, the period of time between detection and response can be months long, and can allow adversaries to attain their goals. To that end, the exploitation of the assigned strength levels and defenders can contribute to the development/utilization of automated incident response techniques, security metrics, and verification properties.

4.1 Strength Levels in HCKC

Throughout the paper, we will assume that the basis of a HCKC is a set S of discrete stages in HCKC. Moreover, E will denote hackers, includes internal and external hackers, which are entities in the model. We now assign the strength level to any entity, which will in practice be related to the degree of strength of its entity.

A lattice for strength concerns, $\mathcal{L} = (L, \leq)$ consists of a set L and a partial order relation $\leq$ such that, for all $l, l' \in L$, there exists a least upper bound $l \sqcup l' \in L$, and a greatest lower bound $l \sqcap l' \in L$. The lattice is complete if each subset L of L has both a least upper bound $\coprod L$ and a greatest lower bound $\prod L$. We will assume that the lattice $\mathcal{L}$ is complete.

A strength level map: $\ell : E \rightarrow L$. This represents the strength level of each hacker.

Firstly, we assign strength levels to stages, which represent the defenders' strength level of each stage:

- $\ell : S \rightarrow L$

Moreover, a new mapping loc is used to return the stage of each entity:

- $loc : E \rightarrow S$

Then we add a rule that an entity can only be deployed in a stage with a strength level that is smaller than that of the entity. That is, for each entity e: an entity e is located at stage s, then we must have

$$\ell(loc(e)) < \ell(e) \tag{1}$$

4.2 A Model for HCKC

We now introduce a formal model for capturing the behaviour of HCKC. The proposed model uses tuples to represent entities in HCKC. Each such tuple comprises information about the nature of the entities (hackers' behaviour). Since there can be duplicates of hackers within the same stage of HCKC, the state of the system is a multiset of entities, allowing for an arbitrary multiplicity of any hackers. The transformations of the system are then defined through the simultaneous execution of individual actions, each action being executed instantaneously and possibly many times.

The system is based on a fixed set of stages. To aid the understanding of the system model, it is introduced in three steps. First, we specify the overall structure in Definition 1. Then, in Definition 2, we introduce rules that explain the transformation between the system states. Finally, we specify the exact format of hackers' actions supported by the model.

Definition 1 (HCKC structure). *A model for Hybrid Cyber Kill Chain is a tuple:*

$$HCKC = (S, E, \mathcal{L}, \ell, \mathcal{A}, st_{init}) \ , \tag{2}$$

where: S is a finite nonempty set of stages*; E is a finite nonempty set of* hackers*; $\mathcal{L}$ is a complete strength lattice; $\ell : S \to \mathcal{L}$ is a mapping assigning strength levels to the stages; $\mathcal{A}$ is a finite set of actions, each action being a pair $\phi = (\phi^{in}, \phi^{out})$ consisting of two finite multisets over the set of tuples*

$$\mathcal{M} = (E \times L \times S);$$

and st_{init} is an initial state defined as a finite multiset over $\mathcal{M}$. In general, a state *of HCKC is a finite multiset over $\mathcal{M}$, and $x \in \mathcal{M}$ is* present *in a state st if $st(x) > 0$.*

A tuple $(e, l, s) \in \mathcal{M}$, denoted by $(e, l)@s$, represents a hacker e with the strength level l at stage s. An entity can have several different copies, and each of these copies can have a different strength level and may appear at different stage. As already mentioned, we allow multiple copies of a single entity to be present in a stage. Hence a state is a multiset st over $\mathcal{M}$ rather than a subset of $\mathcal{M}$. For example, $st_8(e_6, 1, s_4) = 4$ means that in the state st_8 there are four copies of hacker e_6 with strength level 1 at stage s_4.

Now we define how the system can proceed from one state to another state by executing a multiset of actions.

It is assumed that the executed (instances of) actions cannot share input entities. For example, if there is one copy of an entity present, then at most one action which has this entity in its input can be executed. This results in conflicts between actions which could potentially be executed, and contributes to nondeterminism in system execution. The formal semantics, and then property verification, take into account all possible ways in which such conflicts could be resolved.

Below $(-)$ and $(+)$ are respectively the multiset subtraction and addition operations.

Definition 2 (HCKC semantics). *A multiset $\Phi = \{\phi_1, \ldots, \phi_n\}$ of actions over $\mathcal{A}$, where $\phi_i = (\phi_i^{in}, \phi_i^{out})$ for $i = 1, \ldots, n$, is* enabled *at state st if*

$$\Phi^{in} = \phi_1^{in} + \ldots + \phi_n^{in} \leq st \ .$$

Then Φ can then be executed *leading to a state st' given by:*

$$st' = st - \Phi^{in} + \Phi^{out} = st - \Phi^{in} + \phi_1^{out} + \ldots + \phi_n^{out} .$$

We denote this by $st \xrightarrow{\Phi} st'$.

With such a definition we can state precisely what are the states which can be reached from the initial one.

Definition 3 (HCKC reachable states). *The* reachable states *of HCKC in Definition 1 is the minimal set of states RS containing st_{init} such that if $st \in RS$ and $st \xrightarrow{\Phi} st'$, for some multiset of actions Φ, then $st' \in RS$.*

Hackers' Actions Hackers can move between different stages in the chain, so the actions concern the movement of the hackers in different stages. Each action $\phi = (\phi^{in}, \phi^{out}) \in \mathcal{A}$ is such that:

$$\phi^{in} = \{(e,l)@s\} \text{ and } \phi^{out} = \{(e,l)@s'\} , \tag{3}$$

where $s, s' \in S$, $e \in E$, and $l \in L$.

Note also that in practice the actions in $\mathcal{A}$ can be specified in a more convenient way; for example, by using guards and parameters. This is illustrated in the Petri net representation discussed later in this paper, where net transitions use guards and arcs use parameters (variables).

Remark 1. The system model introduced above has been kept deliberately simple. This should allow one to define on top of it a variety of user-friendly, and thus more practical, notations for system specification and property verification. We will demonstrate in the rest of this paper how this can be achieved.

Despite its relative simplicity, the model is very expressive and yet tractable. Basically, it is equivalent to the model of Petri nets introduced in Section 5, where state reachability is decidable.

One could then ask what would happen if we increased the modelling power of the basic system model. A possible extension could allow, for example, to check for the absence of certain kinds of hackers. This would lead, in terms of Petri nets, to the introduction of inhibitor arcs and a loss of the decidability of state reachability as Petri nets extended by inhibitor arcs are Turing powerful. The same would be the case if the execution model assumed that at each step a maximal multiset of action was executed.

Therefore, as one of our aims is to keep the system model tractable, we believe that the formalisation presented above strikes a right balance between being useful for practical applications and amenable to automated verification.

Well-formedness In addition to verifying the property of the model, there are other desirable functional properties which one would normally need to verify using, e.g., model checking tools. The following are examples of such properties formulated for *HCKC* in (2):

- *HCKC* is *live*, if for every reachable state st' and every action ϕ, there is a state st'' reachable from st' at which ϕ can be executed.
- *HCKC* is *bounded*, if there is $n \geq 1$ such that the size of each reachable state is less than n.
- *HCKC* is *well-formed*, if there is a state st such that *HCKC* is both live and bounded after replacing st_{init} by st.

5 Petri Nets

Petri nets are a graphical modelling tool for a formal description of systems whose dynamics are characterized by concurrency, synchronization, mutual exclusion and conflict [13, 18].

5.1 Place/Transition Nets

A Place/Transition net (PTN) N consists of two disjoint finite sets of nodes, Pl and Tr, respectively called *places* and *transitions*, a mapping $W : (Pl \times Tr) \cup (Tr \times Pl) \to \mathbb{N}$ specifying the weights of *arcs* that connect the nodes, and the *initial marking (state)* $M_0 : Pl \to N$. In general, any finite multiset of places is a marking (or state) of N.

Intuitively, places carry (black) *tokens* which represent the current distribution of resources in a system modelled by the net. In other words, the current state of the modelled system is given by the number of tokens in each place.

Transitions are the active components of the net. An input arc of a transition tr starts at a place pl and ends at tr provided that $n = W(pl, tr) > 0$. In such a case, n is the arc's weight signifying that an execution of tr requires n tokens in pl which are consumed as a result. Similarly, an output arc from tr to pl exists provided that $m = W(tr, pl) > 0$, and an execution of tr inserts m tokens into pl.

A transition tr is allowed to be executed (or *fired*) at a marking M if $M(pl) \geq W(pl, tr)$, for all places pl. Its firing produces a new marking M' such that $M'(pl) = M(pl) - W(pl, tr) + W(tr, pl)$, for all places pl. In general, one can fire a finite multiset of transitions $U = \{tr_1, \ldots, tr_k\}$ provided that $M(pl) \geq W(pl, tr_1) + \cdots + W(pl, tr_k)$, for all places pl (that is, input tokens cannot be shared), and its firing results in a marking M' such that $M'(pl) = M(pl) - W(pl, tr_1) - \cdots - W(pl, tr_k) + W(tr_1, pl) + \cdots + W(tr_k, pl)$, for all places pl. One can then consider finite and infinite execution sequences starting from the initial marking, and introduce the notion of a reachable marking.

Petri nets, in particular PTNs, have been widely used for structural modelling of workflows and have been applied in a wide range of qualitative and quantitative analyses (for example, [13, 22, 1]).

5.2 Coloured Petri Nets

PTNs are a low-level model, and in practical applications, it is convenient to use more compact (but behaviourally equivalent) high-level Petri net models. An example of such a compact model are *coloured Petri nets* (CPNs), where the tokens are tuples of values, the arcs are used as selectors allowing one to specify the format of input and output tokens, and transitions have associated *guards* which allow one to easily express, e.g., various security policies.

Let *Tok* be a finite set of elements (or colours) and *VAR* be a disjoint finite set of variable names. In a CPN:

- Each place has a *type*, which is a subset of *Tok* indicating the colour of tokens this place can contain. A marking is obtained by placing in each place a multiset of tokens belonging to the type of the place.
- Each arc is labelled with a multiset of variables from *VAR*.
- Each transition has a *guard*, which is a Boolean expression over $Tok \cup VAR$. For a transition t, $VAR(t)$ denotes the set of variables appearing in its guard and labelling its input and output arcs.

The enabling and firing rules of coloured Petri Nets are as follows: when tokens flow along the incoming arcs of a transition t, they become bound to variables labelling those arcs, forming a binding mapping $\sigma : VAR(t) \to Tok$. If this mapping can be extended to a total mapping σ' in such a way that the guard of t evaluates to *true* and the values of the variables on the outgoing arcs are consistent with the types of the places these arcs point to, then t is *enabled* and σ' is an *enabling binding* of t. An enabled transition can *fire*, consuming the tokens from its pre-set and producing tokens in places in its post-set, in accordance with the values of the variables on the appropriate arcs given by σ'. One can then define an enabling condition and firing rule for a multiset of transitions with enabling bindings similarly as it was done for PTNs, and introduce notions like marking reachability by generalizing those defined for PTNs.

6 Hybrid Cyber Kill Chain in Coloured Petri Nets

We now outline how CPNs could be used to represent (and then used to verify) a given HCKC. We will now use a scenario (Target Data Breach) to illustrate the definition of HCKC model, and the way it can be represented using coloured Petri nets.

In 2013, Target, one of the largest retail companies in the US, was a victim of cyber attack [28]. The attackers surreptitiously managed to gain access to Target's network, stole the financial and personal information of 110 million customers, and then removed this sensitive data from Target's network to a server somewhere in Eastern Europe. Below they are outlined the key points at which Target failed to detect and prevent the attack:

- Target enabled a third-party vendor to have access to their network. The vendor did not follow appropriate information security practices. As a result, the weak security of the vendor allowed the attackers to gain access in Target's network.

- While the attackers were installing malicious software on Target's system, the company failed to respond to a series of automated warnings from their intrusion detection software.
- Attackers managed to move from less sensitive areas of Target's network to areas storing consumer data by leveraging vendor's credentials.
- Target failed to respond to a series of warnings from their intrusion detection software, while the attackers removed Target's data via the network.

For this scenario, we have in total nine stages and one entity (hacker). Stages, $s_i : i \in \{0, ..., 8\}$, are sorted according to the position of each stage in HCKC (see Figure 2), thus s_0 stands for Reconnaissance and s_8 for Action on Objectives. Moreover, based on HCKC, a defence mechanism is mapped to each stage, the purpose is to prevent a hacker to move to next stages in the kill chain. In both, the hacker and defence mechanisms, are assigned a strength level. A higher strength level for a hacker means that is more capable to pass each stage in the kill chain, and for a defence mechanism to block an attack. A hacker can be deployed on different stages.

The strength levels of entities and defence level of different stages are listed in Table 1. In this scenario, the hacker initially attacked the vendor's weak network system and used it to gain access in Target's network. For this reason, two kill chains are combined together to describe how Target's cyber attack evolved. Stage s_1, ... , s_8 belong to the kill chain for the vendor's attack, while for the Target's attack we have the stages s'_4, ... , s'_8. That is because the hacker's aim was to reach the Lateral Movement in vendor's kill chain (i.e., s_7). Then, the attacker gained access to Target's network. The kill chain of the attack, when the attacker managed to enter the Target's network starts from stage s'_4 (i.e., Exploit), as the attacker was already in the Target's network (the previous stages are applied once an attacker is outside the network). The hacker can pass to a next stage only if he/she has higher strength level than a defence mechanism. Here, as the hacker executed a successful cyber attack, the value of his/her strength level is higher ($l = 2$) than the defence mechanisms of all stages.

Table 2 shows all the valid mappings of entities to stages. We can observe, e.g., that e_0 can be deployed at stage $s_1, \ldots, s_8, s'_4, \ldots, s'_8$.

The structure and dynamics of the system are represented by the coloured Petri net in Figure 3. In this diagram, tokens represent entities; places represent different stages; and transitions capture the activities. `s`, `l`, etc., are parameters (variables).

We assume that in the initial state of the system there is one hacker e_0 residing at stage s_0. The hacker is represented by a token in place labelled as s_0 being $(e_0, 2)@s_0$. This net shows how the hacker can move between different stages. Note that the rule for hacker's movement is represented by the guards of the form $l > \ell(s_1)$ associated with the transitions in the net. If the hacker's strength level is higher than the defence level of next stage, the hacker can move to the next stage, otherwise, the hacker can not proceed, the hacker would go back to any previous stages.

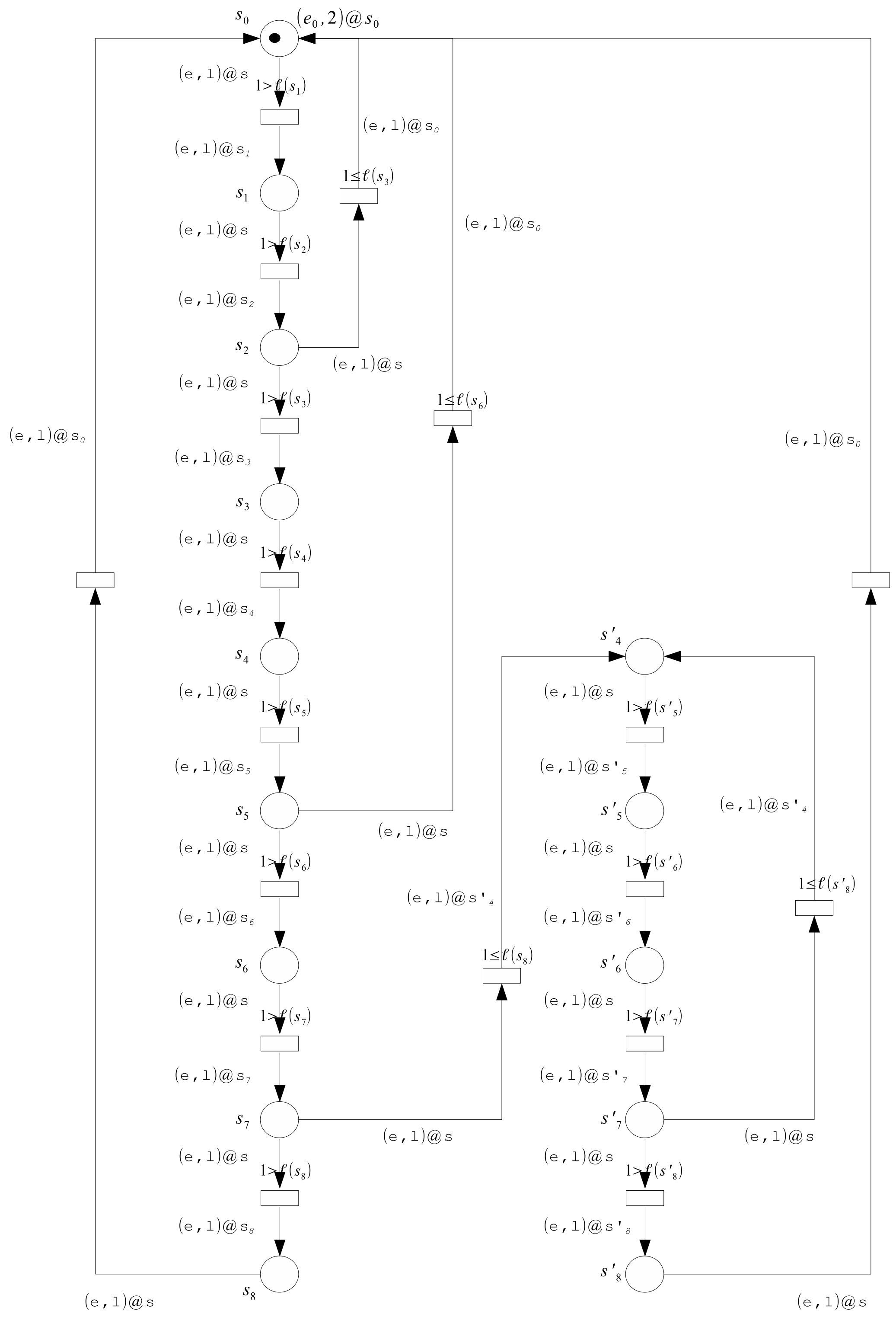

Fig. 3. A Petri net model of a system consists of nine stages (fourteen places) and one hacker, e_0 which resides on stage s_0. Note that places are represented by circles, transitions by boxes.

Table 1. Strength of hackers and defence in HCKC.

Entities	Strength level	Stages	Strength level
e_0	2	s_0	0
		s_1	0
		s_2	1
		s_3	1
		s_4, $s_4^{'}$	1
		s_5, $s_5^{'}$	1
		s_6, $s_6^{'}$	1
		s_7, $s_7^{'}$	1
		s_8, $s_8^{'}$	1

Table 2. Valid mappings of entities to stages

Entities	s_0	s_1	s_2	s_3	$s_4, s_4^{'}$	$s_5, s_5^{'}$	$s_6, s_6^{'}$	$s_7, s_7^{'}$	$s_8, s_8^{'}$
e_0	●	●	●	●	●	●	●	●	●

In general, a CPN model like that outlined above can be translated into a behaviourally equivalent model as in (2) (essentially, each transition tr is replaced by a set of actions obtained from the enabling bindings of tr). Similarly, each system model as in (2) can be translated into a behaviourally equivalent CPN. It is then possible to apply model checking tools developed for CPNs to verify security properties in HCKC.

6.1 Case Study Evaluation

As mentioned in Section 4, HCKC is not only useful for analyzing attacks, but can provide crucial information to system architects and security analysts to make trade-off decisions involving system security. Target's Achilles heel was the weak security of a third-party vendor. The attacker exploited this weakness and managed to gain access to Target's network. In addition, another vulnerability of Target was the failure to respond to the incident as soon as possible. Formal modelling Target's system in advance, using the HCKC framework security analysts could consider that a threat can appear indirectly via third-party interactions, according to Lateral Movement stage in the kill chain.

Moreover, the exploitation of the assigned strength levels and defenders can help the analysts to decide where to apply automated incident response techniques. To this point, we should mention that not all adversaries are alike. Adversaries can be individuals - insiders, outsiders, trusted/privileged insiders; groups - ad hoc or established; organizations - competitors, suppliers, partners, customers; and nation-state - hackers employed by a national government. Different adversaries can aim at attacking different business processes of an organization, and these attacks can happen concurrently. In this case a Petri net model, like the one in Figure 3, can be extended to model these cases. We should mention that the strength levels of entities make possible the formal verification of such complex and highly concurrent systems.

An alternative approach, instead of the security policies based on strength levels, could be the adoption of weak diagnosis [8] and its associated verification property the weakly fair diagnosability [7]. This formal verification technique can detect abnormal behaviours of a system (e.g., cyber attacks). This approach has been tested in [34] and showed that the detection of an adversary becomes more “expensive" in verification time. That is, because the state space increases dramatically when the number of entities become larger.

Finally, it should be noted that all organizations do not have formal models of their business processes. To employ model-based security analysis techniques requires an investment from the organizations. Moreover, overhead costs are incurred to maintain and optimize the formal models over time by experts. However, the utilization of model-based security analysis techniques can provide a return on investment as organizations squandering significant financial and human resources on programs that are not efficient to help them diminish security risks in their shared ecosystems.

7 Conclusion

In this paper, we developed a new framework - HCKC - to analyse how attackers conduct the activities to organizations. Strength level of attackers and defenders are introduced to the framework. A formal model is built to analyse the framework, which can be captured by coloured Petri nets. This study can be used to help security professionals find out the fundamental flaws of the security strategies of organizations. In addition, this study can help security officers take actions to deal with security incidents to eliminate or minimize the impact, and establish strong controls and countermeasures in the business organizations. In the future, we plan to produce model-based security metrics for HCKC. Thus, system architects and Cyber Security Information Officers can use these metrics to design and embed cost-effective security solutions to their systems.

8 Acknowledgement

The authors are very grateful to the anonymous reviewers for their detailed and constructive comments and suggestions.

References

1. van der Aalst, W.M.P.: The application of petri nets to workflow management. The Journal of Circuits, Systems and Computers **8**, 21–66 (1998)
2. Bryans, J.W., Koutny, M., Mazare, L., Ryan, P.Y.A.: Opacity generalised to transition systems. Int. J. Inf. Sec. **7**(6), 421–435 (2008)
3. Bryans, J.W., Koutny, M., Ryan, P.Y.A.: Modelling opacity using petri nets. Electr. Notes Theor. Comput. Sci. **121**, 101–115 (2005)

4. Chen, T.M., Sanchez-Aarnoutse, J.C., Buford, J.: Petri net modeling of cyber-physical attacks on smart grid. IEEE Transactions on Smart Grid **2**(4), 741–749 (2011)
5. Death, D.: The Cyber Kill Chain Explained (2018), `https://www.forbes.com/sites/forbestechcouncil/2018/10/05/the-cyber-kill-chain-explained/#1f349fd06bdf`
6. Engel, G.: Deconstructing The Cyber Kill Chain (2014), `https://www.darkreading.com/attacks-breaches/deconstructing-the-cyber-kill-chain/a/d-id/1317542`
7. Germanos, V., Haar, S., Khomenko, V., Schwoon, S.: Diagnosability under weak fairness. ACM Trans. Embed. Comput. Syst. **14**(4), 69:1–69:19 (2015)
8. Haar, S., Benveniste, A., Fabre, E., Jard, C.: Partial order diagnosability of discrete event systems using petri net unfoldings. In: 42nd IEEE International Conference on Decision and Control. vol. 4, pp. 3748–3753 (2003)
9. Jasiul, B., Szpyrka, M., Sliwa, J.: Malware behavior modeling with colored petri nets. In: Computer Information Systems and Industrial Management. pp. 667–679. Springer Berlin Heidelberg, Berlin, Heidelberg (2014)
10. Kristensen, L.M., Simonsen, K.I.F.: Applications of coloured petri nets for functional validation of protocol designs. T. Petri Nets and Other Models of Concurrency **7**, 56–115 (2013)
11. Laliberte, M.: A Twist On the Cyber Kill Chain: Defending Against a JavaScript Malware Attack. Dark Reading (2017)
12. M., H.E., Cloppert, M.J., Amin, R.M.: Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. Bethesda, MD: Lockheed Martin Corporation (2010)
13. Marsan, M.A., Balbo, G., Conte, G., Donatelli, S., Franceschinis, G.: Modelling with generalized stochastic Petri Nets. Wiley Series on Parallel Computing (1995)
14. Mason, S.: Leveraging the kill chain for awesome. DARKReading (2014)
15. Mason, S.: Leveraging the Kill Chain for Awesome (2014), `https://www.darkreading.com/attacks-breaches/leveraging-the-kill-chain-for-awesome/a/d-id/1317810`
16. McDermott, J.P.: Attack net penetration testing. In: Proceedings of the 2000 Workshop on New Security Paradigms. pp. 15–21. NSPW '00, ACM, New York, NY, USA (2000)
17. MITRE: MITRE Research Opens Window into Cyber Attacker Behaviour. The MITRE Corporation (2015), `https://www.mitre.org/news/press-releases/mitre-research-opens-window-into-cyber-attacker-behavior/`
18. Murata, T.: Petri nets: Properties, analysis and applications. Proceedings of the IEEE **77**(4), 541–580 (1989)
19. Myers, L.: The practicality of the cyber kill chain approach to security. CSOOnline (2013)
20. Nachreiner, C.: Kill Chain 3.0: Update the cyber kill chain for better defense (2015), `https://www.helpnetsecurity.com/2015/02/10/kill-chain-30-update-the-cyber-kill-chain-for-better-defense/`
21. Reidy, P.: Combating the Insider Threat at the FBI. Presented at Blackhat USA (2013), `https://media.blackhat.com/us-13/US-13-Reidy-Combating-the-Insider-Threat-At-The-FBI-Slides.pdf`
22. Salimifard, K., Wright, M.: Petri net-based modelling of workflow systems: An overview. European Journal of Operational Research **134**(3), 664–676 (2001)
23. Schneier, B.: Secrets & Lies: Digital Security in a Networked World. John Wiley & Sons, Inc., New York, NY, USA, 1st edn. (2000)

24. Sheyner, O., Haines, J.W., Jha, S., Lippmann, R., Wing, J.M.: Automated generation and analysis of attack graphs. In: 2002 IEEE Symposium on Security and Privacy, Berkeley, California, USA, May 12-15, 2002. pp. 273 – 284 (2002)
25. Sheyner, O.M.: Scenario Graphs and Attack Graphs. PhD Thesis, Carnegie Mellon University, (2004)
26. Simonsen, K.I.F., Kristensen, L.M., Kindler, E.: Pragmatics annotated coloured petri nets for protocol software generation and verification. T. Petri Nets and Other Models of Concurrency **11**, 1–27 (2016)
27. Stephenson, P.R.: A formal model for information risk analysis using colored petri nets. In: Proceedings of the Fifth Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools. pp. 167–184 (2004)
28. US Senate Committee on Commerce, Science and Transportation: A 'Kill Chain' Analysis of the 2013 Target Data Breach (2014), `https://www.commerce.senate.gov/public/_cache/files/24d3c229-4f2f-405d-b8db-a3a67f183883/23E30AA955B5C00FE57CFD709621592C.2014-0325-target-kill-chain-analysis.pdf`
29. Wu, R., Li, W., Huang, H.: An attack modeling based on hierarchical colored petri nets. In: Proceedings of the 2008 International Conference on Computer and Electrical Engineering. pp. 918–921. ICCEE '08, IEEE Computer Society, Washington, DC, USA (2008)
30. Xu, D., Nygard, K.E.: Threat-driven modeling and verification of secure software using aspect-oriented petri nets. IEEE Transactions on Software Engineering **32**(4), 265–278 (2006)
31. Zeng, W.: A methodology for cost-benefit analysis of information security technologies. Concurrency and Computation: Practice and Experience **31**(4), e5004 (2019)
32. Zeng, W., Germanos, V.: Benefit and cost of cloud computing security. Harnessed Causality: Essays Dedicated to Maciej Koutny on the Occasion of His 60th Birthday pp. 143–150 (2018)
33. Zeng, W., Koutny, M.: Data resources in dynamic environments. In: 2014 Theoretical Aspects of Software Engineering Conference. pp. 185–192 (2014)
34. Zeng, W., Koutny, M., Watson, P., Germanos, V.: Formal verification of secure information flow in cloud computing. J. Inf. Sec. Appl. **27-28**, 103–116 (2016)