- Q1. We would like to combine the Cyber Kill Chain (CKC) Model with our misuse patterns. Describe the idea and use as example the DDoS attack using Botnets or a similar attack. Bonus: combine misuse patterns with SAMIIT to get a misuse pattern for CPS and IoT.

Combining misuse patterns with the Cyber Kill Chain (CKC) presents a few difficulties. The Cyber Kill Chain[1], a Lockeed-Martin trademark, and SAMIIT (Spiral Attach Model in Industrial Internet of Things)[2] are attack models showing the steps an intruder will take to compormise a system and assist the planner interrupt the attack at any stage. The Cyber Kill Chain breaks the attack into distinct linear phases. SAMIIT further subdivides the attack into Domains and restarts the attack and refines the phases of attack into External/Internal/Target, where each phase repeats the CKC attack sequences. The spiral pattern occurs when the attack has penetrated the IT to DMZ zone and is exploiting internal flaws within the system. Misuse patterns are intended as an overall view of an attack and are designed to allow an system designer to design systems with minimal exposure.

While CKC and SAMIIT allow detection and interdiction, pattern is designed for prevention. The detection of the CKC and SAMIIT are implicitly contained in the Misuse Pattern under the Countermeasures and Related Patterns sections. The CKC and SAMIIT are explicitly covered in the pattern when the misuse pattern contains a sequence diagram. Using the *Internet of Things DDoS White Paper* from the Electricity-Information Sharing Information Center (EISIC)[3] and the class notes from the IoT DDoS sequence diagram, I will attempt to map the Mirai DDoS attack from 2016 to the phases in CKC/SAMIIT[4]. The attack process for an IoT device is similar to that of a larger device. The reader is directed to compare the sequence diagram below to sequence diagram 4 from *A Worm Misuse Pattern*[5] (the worm and IoT infection work in similar patterns, I will map the sequence of actions to CKC and SAMIIT. The Worm Misuse pattern was chosen because it is well known and exemplifies the steps used in a DDoS attack.

CKC	***SAMIIT***		***Misuse Pattern Step***	***Mirai***	***Notes***
Reconnaissance	IT Domain L5 & L4	External Reconnaissance	1a-d: Setup	SYN to pseudo-random IP addresses ports 23/2323	
			2: Use brute force to get credentials	Attempt Brute force login (64 predefined	See table 5 from footnote 4 reproduced

[1] Eric Hutchins, Michael Cloppert, Rohan Amin. Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains, 2010. URL: https://isgs-gen.external.lmco.com/sites/ECS/Marketing/White%20Papers%20and%20External%20Publications/Cyber%20Kill%20Chain%20Whitepaper.pdf, last accessed 9-Nov-2018

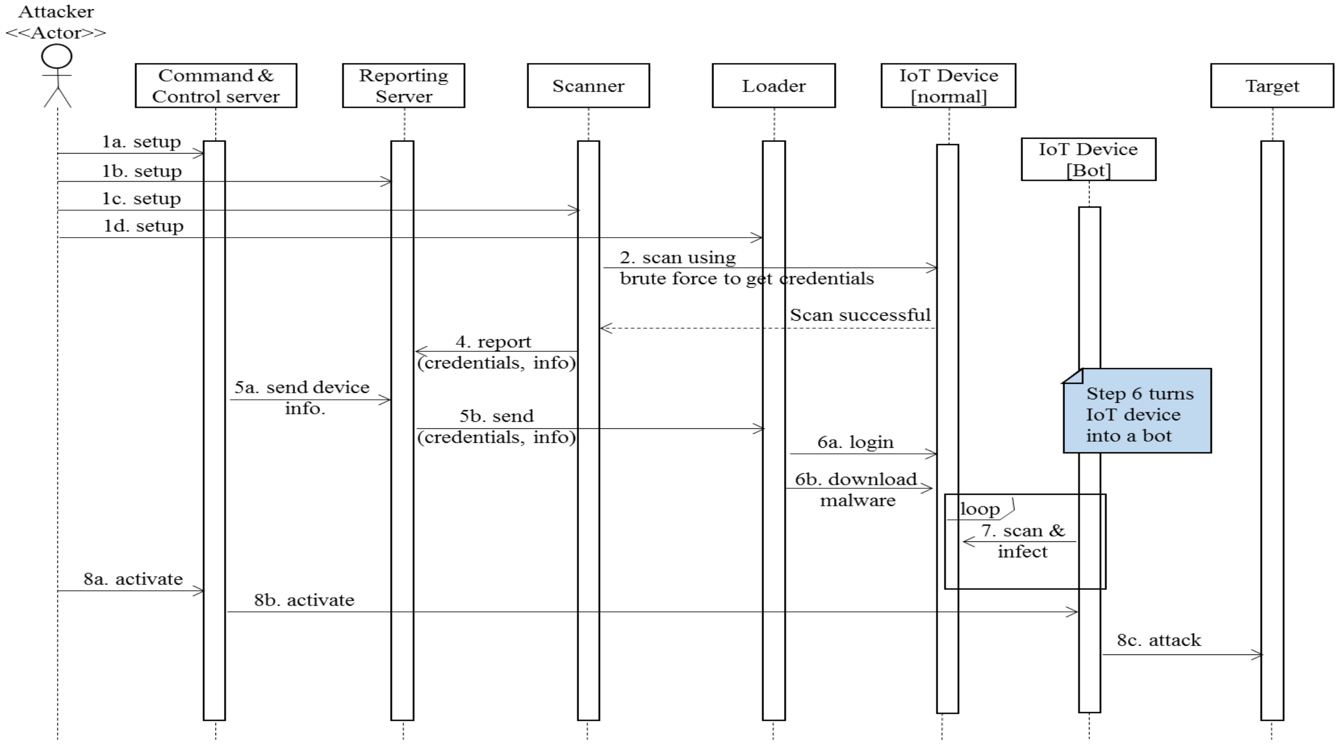
[2] Amin Hassanzadeh, Robin Burkett, SAMIIT: Spiral Attack Model in IioT Mapping Security Alerts to Attack Life Cycle Phases, DOI: http://dx.doi.org/10.14236/ewic/ICS2018.2

[3] *Internet of Things Whitepaper*,Electricity-Information Sharing and Analysis Center, https://c.ymcdn.com/sites/members.iamu.org/resource/resmgr/informer_2016/IOT.pdf last accessed 9-Nov-2018

[4] M. Antonakakis, et al., *Understanding Mirai Botnet*, https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/46301.pdf, last accessed 9-Nov-2018.

[5] E.B. Fernandez, N. Yoshioka, H. Washizaki, *A Worm Misuse Pattern*

				user/password combinations)	below.
Weaponization		Ext. Weaponization	4: Report compromised machine	Send IP and login to *report server*	
Delivery		Ext. Delivery	5: Send device info, credentials	Login and determine system structure	
Exploitation		Ext. Exploitation	6a: Login	Download architecture specific loader	
Installation		Ext. Installation	6b: download Malware	Install and execute	Malware may be architecture specific.
Command and Control (CnC)		External CnC	7: Loop scanning and infecting as per 1-6	Kill processes bound to ports 22/23	Compromised system acts as scanner, until execute command given.
				Kill processes linked to mirai variants	
Act on Target	IT to DMZ and OT Levels 5 - 2	Internal Reconnaissance	2: Scan use brute force login	Scan for other victims	IoT systems may/not require layered attack depending on security breach. (eg. compromised user vs compromised admin account).
		Int. Weaponization	4: Report vulnerabilities		
		Int. Delivery	5: Send credentials		
		Int. Exploitation	6: Login remote system		
		Int. Installation	6b: Download		
		Int. CnC		Listen for command from Master	
	OT Control Levels 2-0	Target Reconnaissance		Command Received	
		Target Weaponization			
		Target Delivery			
		Target Explolit			
		Target Installation			
		Execute	8c: attack		



Password	Device Type	Password	Device Type	Password	Device Type
123456	ACTi IP Camera	klv1234	HiSilicon IP Camera	1111	Xerox Printer
anko	ANKO Products DVR	jvbzd	HiSilicon IP Camera	Zte521	ZTE Router
pass	Axis IP Camera	admin	IPX-DDK Network Camera	1234	Unknown
888888	Dahua DVR	system	IQinVision Cameras	12345	Unknown
666666	Dahua DVR	meinsm	Mobotix Network Camera	admin1234	Unknown
vizxv	Dahua IP Camera	54321	Packet8 VOIP Phone	default	Unknown
7ujMko0vizxv	Dahua IP Camera	00000000	Panasonic Printer	fucker	Unknown
7ujMko0admin	Dahua IP Camera	realtek	RealTek Routers	guest	Unknown
666666	Dahua IP Camera	1111111	Samsung IP Camera	password	Unknown
dreambox	Dreambox TV Receiver	xmhdipc	Shenzhen Anran Camera	root	Unknown
juantech	Guangzhou Juan Optical	smcadmin	SMC Routers	service	Unknown
xc3511	H.264 Chinese DVR	ikwb	Toshiba Network Camera	support	Unknown
OxhlwSG8	HiSilicon IP Camera	ubnt	Ubiquiti AirOS Router	tech	Unknown
cat1029	HiSilicon IP Camera	supervisor	VideoIQ	user	Unknown
hi3518	HiSilicon IP Camera	<none>	Vivotek IP Camera	zlxx.	Unknown
klv123	HiSilicon IP Camera				

Table 5: **Default Passwords**—The 09/30/2016 Mirai source release included 46 unique passwords, some of which were traceable to a device vendor and device type. Mirai primarily targeted IP cameras, DVRs, and consumer routers.

- Q2. Study OpenStack security. Find threats and countermeasures. Look in the Internet, make a structured and systematic list according to architectural effects.

OpenStack is an open source Cloud Computing system which provides IaaS services.[6] With the growth of the internet and the desire of many companies to offer services many turn to Open Source projects. When discussing Open Source security it becomes necessary to separate fact from fiction. As one observation stated "The security discussion of open source and closed source software is rife with beliefs and guesses. Data-driven insights based on an empirical analysis, as examined here, provide new insight into such security issues."[7] Although open source advocates argue that more eyes means more bugs found and better code, the same article also noted an observation that in one early Open Source firewall project that although over 2000 sites were using the firewall only 10 developers actually took time to suggest patches.[8] Major security breaches in recent years were due to improperly patched or maintained Open Source code (Equifax). Other examples could include an image manipulation tool used at sites like dragonmemes.com which had known flaws, but the original developer was no longer interested in maintaining the Open Source code. In its 2017 Report On Open Source Security, SNYX determined that the median lifespan for a flaw is 2.5 years and that 25% of developers do not tell users of security flaws in their code.[9]

It should be noted that the assignment was to discuss Open Stack. Therefore I should begin with a review of data protection in open stack. To protect data, Open Stack provides three features, Key Management, Block Encryption and image integrity.[10] Barbican, the open stack key management service provides an interface for managing keys and secrets. Using appropriate keys and authentication the user can be verified and appropriate access granted. The block storage of data, a target for malevolent users, can exist either as permanent or temporary (virtual machine memory). In either case the ability exists to encrypt and decrypt the data using keys stored/created in the key store. Encryption is user configurable and a shared responsibility is assumed. Encryption of the data will protect the data while in transmission from the data store to the Virtual Machine. This will prevent multiple avenues of attack. However, the ability to encrypt may be disabled by the user. In early version of Open Stack, the integrity was ensured through use of an MD5 checksum. In recent versions, the integrity of images is ensured through use of a signature that is sent separately from the the image. This feature is also user selectable and if no signature is received the image integrity is not checked.

[6] "Security Assessment of OpenStack Cloud Using Outside and inside Software Tools." *2018 International Conference on Development and Application Systems (DAS), Development and Application Systems (DAS), 2018 International Conference On*, 2018, p. 170. *EBSCOhost*, doi:10.1109/DAAS.2018.8396091.

[7] Schryen, Guido. "Is Open Source Security a Myth?" *Communications of the ACM*, vol. 54, no. 5, May 2011, pp. 130–140. *EBSCOhost*, doi:10.1145/1941487.1941516.

[8] "Open Source Security: Opportunity or Oxymoron?" *Computer*, no. 3, 2002, p. 18. *EBSCOhost*, doi:10.1109/2.989921.

[9] "2017 State of Open Source Security", https://snyk.io/stateofossecurity/, last accessed 12-Nov-2018.

[10] "Data Protection in OpenStack." *2017 IEEE 10th International Conference on Cloud Computing (CLOUD), Cloud Computing (CLOUD), 2017 IEEE 10th International Conference on, CLOUD*, 2017, p. 560. *EBSCOhost*, doi:10.1109/CLOUD.2017.77.

Open Stack uses the Keystone module to identify and authenticate users. An analysis of Keystone in June 2017[11] indicated that over the years Keystone has made major efforts to improve its authentication and logging of events. Early releases of Keystone did not properly invalidate tokens which would allow users to obtain elevated privileges. This has been fixed in recent releases. The most recent concern found is a denial of service due to a large number of authentication requests ("authentication chaining").

Finally using Open Source tools, researchers at the University of "Stefan cel Mare" analyzed Open Stack and determined that certain ports and interfaces are vulnerable to attack from open ports[12] they also found a number of anomalies that they considered small security flaws[13]. Other researchers [14] reviewed CVE's, patches and the system interfaces and determined that while Open Stack is secure from many external attacks, 2/3 of the existing vulnerabilities are due to internal flaws. Many are due to simple programming errors that rigorous code review procedures could have discovered.

In reviewing the Security of the Open Stack it appears as if security is addressed at all layers. The security issues and flaws may be outlined as follows:

- Open Ports – these may be secured through the use of proper configuration.
- User Privileges – correctly assigning privileges through Role Base Access Control (RBAC) can prevent user privilege escalation and limit access.
- Access is logged.
- Interfaces require tokens which contain identity and authorization.
- The system is still subject to DDoS attacks.
- Attacks from an internal source are more likely to succeed.
- In their conclusions, Elia et al. conclude that more than 20% of the vulnerabilities remain undiscovered for over a year, a figure consistent with the analysis of Open Source software conducted by references 7, 8 and 9 above.

[11] Keystone GAP and Threat Identification (Quick Study) OpenStack Folsom Release, https://wiki.openstack.org/w/images/c/c9/OpenStack_Keystone_Analysis.pdf

[12] "An Analysis of OpenStack Vulnerabilities." *2017 13th European Dependable Computing Conference (EDCC), Dependable Computing Conference (EDCC), 2017 13th European, EDCC*, 2017, p. 129. *EBSCOhost*, doi:10.1109/EDCC.2017.29.

[13] "Security Assessment of OpenStack Cloud Using Outside and inside Software Tools." *2018 International Conference on Development and Application Systems (DAS), Development and Application Systems (DAS), 2018 International Conference On*, 2018, p. 170. *EBSCOhost*, doi:10.1109/DAAS.2018.8396091.

[14] Ristov, Sasko, et al. "Security Vulnerability Assessment of OpenStack Cloud." *2014 Sixth International Conference on Computational Intelligence, Communication Systems & Networks*, Jan. 2014, p. 95. *EBSCOhost*, ezproxy.fau.edu/login?url=http://search.ebscohost.com/login.aspx?direct=true&AuthType=ip,cookie,url,uid&db=edb&AN=102532599&site=eds-live&scope=site.