5. Hacker-AI - Advanced Features, Considerations

Fortification / Protection of Position

Fortification and protection of gained positions are designed to defend malware against counterattacks that could remove malware or reveal what it has been gained before.

The most important aspect in Cyberwar 2.0 operations is to remain undetected. Without detection, there is no reasonable defensive action available or possible. We already have invisible software. Undetectable (like Cyber Ghosts) is a level of hiding that deserves to be a quality on its own. The same applies to irremovable (Cyber Devils). However, this undetectable and irremovable feature cannot be seen as absolute because, outside the existing framework (defined by the existing cybersecurity tools and underlying paradigms), Cyber Ghosts and Cyber Devils are likely detectable and removable.

Investing in malware and spreading it on systems without direct, current, or immediate use would be seen as a wasted investment. So it would be reasonable to provide technology that could be left behind, like a private backdoor that could not be removed, allowing attackers to come back later easily. Instead of building a Cyber Beachhead and Cradle again while finding new rights permission methods and the right elevation, the private backdoor would allow the attacker to get in once-visited devices despite users updating their OS or software.

(11) Cyber Ghosts

Software is already invisible; it runs in the background without interrupting users or disturbing them with notifications. But this property alone is insufficient to make software or malware a Cyber Ghost. Background processes on UNIX/Linux, a popular server OS system, are called demons. Administrators, security experts, and even normal users can easily see background processes or demons because they are managed and displayed by all multitasking OS implementations. Suppressing this output would require in-depth OS modifications.

I have defined Cyber Ghosts as undetectable software with tools provided by (existing) cybersecurity. I interpret undetectability literally as the ability of these ghosts to persistently and permanently evade detection from security software.

There are (at least) five basic discovery methods to detect hidden processes on a computer. Cyber Ghosts must avoid them all: (1) The OS has built-in tools for displaying a full list of running processes like Window's Task Manager or Mac's Activity Monitor. (2) We can check on windows systems in about 50 known/supported locations where software can be started automatically without confirmation. All OS have comparable known locations. (3) Security tools could check where software instructions are stored (suspicious files or directories); malware could also be kept where only firmware or BIOS is usually stored. (4) Hidden processes could also be found in RAM or encrypted within RAM, e.g., in instances that run legitimate processes in virtual machines while RAM is inspected. (5) Hidden malware could also create or leave insufficiently removed traces of its presence in storage media, RAM, or higher-level cache memory.

Before I come to the cat-and-mouse game around Cyber Ghosts, there is an open question: if security detection methods analyze a hidden malware process, would they flag it as hidden malware or ignore it? Would security tools detect malware when it is changed or camouflaged? This question is related to the underlying methodology of detecting malware: looking for known malware on a blacklist, i.e., known patterns managed in blacklists, or are there criteria that security software uses and could have kept for itself and from being known by Hacker-AI? Could these criteria be detected, tricked, and spoofed?

Alternatively, if security software looks at whitelists of known, inconspicuous software being accepted to run on systems, then this software could accept known, trusted applications while focusing on the unknown apps and testing them more thoroughly to determine if they have malware features. But what will it do? Dynamic software could already duplicate and modify itself and slip away; where is it next? Nothing tangible in software could make security software able to grab and hold onto software, even if it thinks it detected something. Moreover, it is unlikely that security software knows how the malware is being called or under which name (if any) it does appear in the process list or filesystem. Catching it is useless if the investigated software is not immediately frozen or deactivated. If active, it can move away. But how can it be frozen if the mechanism to give that malware CPU time is not understood?

Security tests are usually not the only software operation run on a multitasking computer. Starting a system with a single app is not a problem, but this is likely not enough. This app would have serious limitations. Due to the variety and complexity of how systems start, we defenders cannot even be sure if the Cyber Ghost has created a virtual machine for this single security app. We depend on many layers of technical complexity in multitasking systems, and we assume that they behave as originally designed, developed, and deployed. Systems may appear normal, but they are already compromised and under malware control. Cybersecurity has no tools (to my knowledge) to convince us sufficiently that we are not deceived by some compromised software running in the background. Without reliable, more advanced detection tools, we are blind and unaware that bad things happen on our devices.

We should assume that security checks are just one (or a few) among many on systems with multiple concurrent processes. Security software (like a cat) will try to chase malware (the mouse). Security software is going systematically or even unpredictably through validation tasks. If a Cyber Ghost is still active, we must assume that it will counter all actions by the security software, i.e., the mouse will adapt to the cat. The Cyber Ghost will likely know what files or folders have been checked already; it could adapt, i.e., move its hidden files back into checked folders covertly while preventing event messages from being issued. The same could happen to other detection methods.

Normally, operating software changes internal file/data structures and reveals (thereby) its existence. Malware would likely not use the OS versions of read, write or delete operations; its own code doesn't create event notifications or other traces. How could malware suppress that it has modified the OS? It modifies the OS on the filesystem before the modified files are checked, or it creates a duplicate that is being checked instead. The malware could hide in memory blocks marked damaged, although undamaged. And in RAM, it creates files or records for the security software in a second version of that software as if everything is OK. Consider the just presented hiding methods only the tip of an iceberg. The OS is highly optimized; this opens many doors for deceiving the security software.

Cyber Ghosts do not need to hide their entire code; it is sufficient that there is a loader that can load code and bootstrap malware into a position of superior control. Some assume this is impossible because loading must happen in a certain sequence - that is what they know and do if they start software. They argue a super-controlling software is the first floor of a tower; it needs to be started first, and only that way it respects the right sequence, i.e., building the next floor on a lower floor. They argue only this will guarantee the desired level of control as everything on the next or upper floor(s) must pass through the lower floor(s); additionally, they may say we are dealing with a running system. Therefore, injecting controlling code between these floors or layers, i.e., into an early/low level/layer, might be difficult - but they think it's impossible. I don't see evidence for that. If low-level injections in running code are feasible, the Cyber Ghost could hide in any microcontroller/firmware and wait for certain criteria to create a restart signal. After it restarted, it could regain its position of superior control.

Could injecting into a superior position also be done by security software? Theoretically, yes. But it may be too late because the Cyber Ghost via the remote Hacker-AI would have detected that, modified the security software, and removed that feature. The problem is that Cyber Ghost-like malware can deceive and manipulate security software anytime. After all, security software was installed on a machine or made accessible before being started on the device on which the Cyber Ghost is hiding.

Cyber Ghost could find many methods to delay the execution of the security software, including a blue screen of death. However, low-level security software requires low-level installation, so there is likely plenty of time to do something by the Cyber Ghost to be prepared for the security software doing its scanning. If necessary, the Cyber Ghost could run the security software in a local simulator, record the output, and replay it to users while the actual security software was stopped and deactivated early. It is replaced with software that only provides expected harmless output but never reveals that it found something suspicious.

Some argue that Hacker-AI will need keys for OS updates; however, it could steal, use/generate them covertly. Alternatively, malware could modify software components that require the validation of update certifications; it could lie to every component and claim it has successfully validated the certificate. Every test can be bypassed, or the component stopping the update process could be deactivated.

Cyberdefenders know about all mentioned weaknesses. They consider it an arms race between the detection of modifications and the modification of detection. Who wins? The Hacker-AI - because the ghost can always analyze the latest OS or security software version. The cyberdefender cannot win when the attacker remains observant and vigilant toward easily detectable changes to the OS. Hacker-AI must also be fast enough to simulate challenges provided by the security software, adapt to new challenges, and automatically deliver workarounds that do not endanger malware's existence or mission.

Even security software that monitors or checks for side-channel data that could indirectly indicate suspicious malware activities is not new. The Cyber Ghost malware is likely being made aware of this via Hacker-AI and prepared to respond automatically. All the ghost must do is disguise its data or activities by giving cyberdefenders more conservative or conventional explanations.

The only chance of successfully detecting or removing ghosts would be an external audit of the storage media (hard drive, SSD, etc.) and a reset of all firmware. What would we find if external devices that detect ghosts are already compromised? Our current cybersecurity approach cannot detect or remove Cyber Ghosts with certainty.

The key property of Cyber Ghosts is undetectability. The implementation of this feature will change in an arms race with defenders. There is no formal proof that the Cyber Ghost operator has an advantage if the required features are correctly implemented. Within the current paradigms of cybersecurity solutions and the limited detection capabilities of traditional security measures, it is hard to see how defenders can gain a sustainable advantage over Cyber Ghosts; however, nothing less is required to stop malware.

(12) Cyber Devils

Some would argue that something already undetectable, i.e., a Cyber Ghost, cannot be removed. For human engineers, this statement might be true. But we have no evidence yet to make these types of inferences from what undetectable software could imply. Undetectable software will likely be difficult to remove.

Let's assume there is a second Cyber Ghost on a system. They may not see or detect each other, but there might be events triggered by a Cyber Ghost creating data that it must remove as data traces later. The other ghost could pick up this data because it has features likely unknown to the other ghost. Having unknown detection features is the main difference to security software, from which we assume their details are known and well-understood by Hacker-AI.

Depending on how each of these Cyber Ghost's versions is implemented, one of these ghosts will likely have more control over the entire CPU. The reason for this argument is that a multitasking OS is layered vertically - like floors in a tower. The next/upper layer always calls a single program in the lowest layer. No second parallel program on the lowest layer could jump in and start a parallel app when needed. Parallel execution, e.g., on another core, would likely be detected by the main (lowest-level) program and then killed (detectable apps cannot be ghosts). This root of all could (theoretically) extend its control to other parallel operating components like firmware running on microcontrollers of devices (storage, network, audio, or video).

A Cyber Ghost is a Cyber Devil when it is irremovable. But so far, the Cyber Ghost is software, meaning it is mutable and, therefore, likely removable. Could a Cyber Ghost make itself immutable? Probably not. To get to irremovable software, we must have another feature that I call dominance. Dominance will relentlessly fight to be in the first layer and control the entire device.

So, being invisible and dominant does not mean that the Cyber Devil processes are deeply evil, immediately malicious, or even a supernatural entity. Still, with this name choice, I want to express that a Cyber Devil is invisible, undetectable, and much worse than a Cyber Ghost (or demons, i.e., background processes) to remove. Still, the Cyber Devil would dominate the hardware and its associated software; it could have hidden, nasty, dangerous, or overwriting capabilities. This Cyber Devil would fight off any late-coming Ghost that would challenge it for being dominant. It is permanent as long as it is not challenged with a new security paradigm.

The Cyber Devil is likely adaptable and mutable in its components. However, irremovable means even reinstalling the OS, including formatting the storage media, would not remove this Cyber Devil. Some associate these capabilities with rootkits and call software with these features rootkits. But irremovability is a quality with its own weight; it should therefore be named separate from rootkit features, i.e., software using low-level sysadmin features.

Additionally, I assume from a Cyber Devil that it would sooner or later get distributed to all devices and establishes footholds everywhere. Getting on all devices is not a hurdle for Cyber Ghosts coming from Hacker-AI, as we assume that adapting to different device types, CPUs, or OS/software is already an assumed feature.

A Cyber Devil should be concerned about another Cyber Ghost created from Hacker-AI operated by a different master/operator. This could lead to a fight between malware generated by different Hacker-AI versions for dominance on every challenged device. I assume that there is a strong incentive to remain dominant. The more dominant malware generated by Hacker-AI could fight off all late-comers and establish itself as the only one; this first and possible only malware platform on a system would then be the Cyber Devil.

This Cyber Devil is likely a low-level ultra-hypervisor (i.e., a super-supervisor) that protects itself from being removed by any update, re-installation, and even reformatting of the hard drive. When the system starts fresh from an uncompromised drive, USB, or DVD, this doesn't mean that an ultra-hypervisor was not started before. Hacker-AI could involve BIOS/UEFI or firmware code to be interjected within a fresh installation or start. To my knowledge, this kind of software doesn't exist yet, but it would be wrong to exclude the possibility that someone develops this kind of malware as a cyberweapon.

However, when Cyber Devils are created and improved by Hacker-AI, they could likely be updated; this could mean that their operators could remove them. Some people at the top and potentially some of the architects will have the knowledge and the keys to modify the Hacker-AI manually so it could behave differently. Some engineers and operators will (literally) control humankind's destiny by controlling Hacker-AI.

From Cyber Ghosts, I assume that they detect newly installed software and stop them from being started until Hacker-AI has analyzed the new software. The same here: if it is too late to stop a late-coming malware and it starts making low-level changes, then I can imagine that this Cyber Devil would reinstall or restore itself from multiple different hidden locations. In that case, it could start playing cat and mouse with the newcomer while having a slight advantage because it would already be on that system and prepare for this contingency with traps and hidden dead drops.

Irremovable is the key feature of Cyber Devils that Hacker-AI could continuously improve against late-comers; basic features for doing so are already included in Cyber Ghosts. But Cyber Devils must be able to defend themselves enough to resist deactivation or removal when they are not in touch with a remote Hacker-AI.

(13) Covert/Private Backdoor Facilitator

Backdoors are already private, covert, intentionally hidden, or disguised, and stealthy, i.e., they avoid detection by reducing visibility or detectability. But if these backdoors are discovered for any reason, they could immediately be misused at scale. Without protection, the entire malware system could be taken over. Therefore, Hacker-AI must protect these backdoors; otherwise, there could be uncontrolled unauthorized access without detection.

Covert doors are already designed to blend in with their surroundings. They are usually equipped with additional security measures. But normal password protection is insufficient because code can be analyzed, and all access restrictions can be removed via reverse code engineering. It doesn't matter how difficult backdoors are hidden. Confronted by automated, AI-based systems, it is not enough to have them disguised as legitimate code.

Hacker-AI would probably use asymmetric backdoors, a solution based on asymmetric cryptography, also known as public-key cryptography. Asymmetric cryptography uses two keys, one for encryption and another for decryption; one is public, and the other is private. The public key is used to encrypt messages, and the private key is used to decrypt them. In an asymmetric backdoor system, private and public keys are kept secret so only the malware operator can message backdoors, which then releases a local encrypted (secret) key that decrypts code that triggers further instructions.

With this system, the covert/private backdoor could only be opened and activated by someone who created the backdoor. These concepts are discussed in kleptography, a cryptography branch dedicated to creating systems and techniques to steal or tamper with data secretly. These cryptographic backdoors and their protocols allow attackers to access systems they once occupied but kept unused. Asymmetric backdoors could be implemented in groups; they could wake up regularly and check on each other if any of them have been removed via updates. Being in a group would make it less likely that all are removed from the operation. As soon as one is missed, a new one could be created.

The insertion of hidden covert, private, but asymmetric backdoors allows an attacker to access the system. These backdoors could be activated via, e.g., a temporary file or cookie downloaded by browsers stored in the browser cache or cookie DB. This dead drop system can be designed so backdoors can be activated without being detected.

The goal is to keep an inconspicuous presence on a machine that was being hacked before so that the attacker can go back without exposing itself to detection from, e.g., suspicious reduction in performance or other indicators. Hacking a system and gaining sysadmin rights is potentially an effort that a Hacker-AI operator wants to build on and not give up due to software updates or the risk of removal via security software. The backdoor technology must be updated when security software has detected them.

It is conceivable that this backdoor protection also has a component that regularly checks for threats from security scans and updates in security tools. This component would help keep the backdoors undetected by removing them traceless and reestablishing them (traceless) after the scan.

The key property of this feature is that if the backdoor is found, it cannot be misused. Implementing this feature could potentially lead to an arms race with defenders. Still, Hacker-AI operators, who want to grow their base of devices based on past successes, need a feature like this.

Misdirection/ Decision Layer

Misdirection and deception mislead or distract adversaries; they influence how other actors (nations, the public) perceive the use of cyberweapons. The goal of deception is to create confusion or uncertainty in adversaries' assumptions, making it difficult for them to respond effectively to a threat or challenge. Presenting a (credible) patsy could serve this purpose. Victims are seeking explanations for having closure with extraordinary or unexplained events. Attackers are not interested that cyberdefenders spending resources to endanger their secrets. With misdirection, attackers control the narrative, i.e., how Hacker-AI or Cyberwar 2.0 events are interpreted.

Decision-making, on the other hand, refers to evaluating options and their execution based on gathered information and desired outcomes. Cyberwar involves a wide range of actions and tactics, making it a complex and dynamic situation with many parallel circumstances that are changing and evolving independently. This dynamic environment is monitored with tools. Detected changes can be returned as feedback.

(14) Cyber Patsy Designer

Humans need answers or explanations if something curious, bad, or big happens. Without a mainstream, dominant story, someone else will come up with narratives that connect the dots of the few facts that are publicly known, potentially in a more damaging way. Which narrative prevails in our culture is almost unpredictable, but which is chosen by opinion leaders is relevant. "Cui Bono", who benefits, is within politics often the starting point of speculation without hard evidence. But also, who has the tools or capabilities or has a credible motive, if the tools/capabilities are comparably easy to come by, is being used to name a culprit. Another popular way to find culprits is to follow the tracks of money.

However, public officials or someone with legal accountability should make decisions based on facts and not speculation. In investigations of crimes or legal matters, forensics is used in which scientific methods and techniques are used to examine and analyze physical evidence, like fingerprints, DNA, or other trace materials, to establish facts or circumstances of a particular case. Forensic evidence is used to establish guilt or innocence or to support an argument or claim.

In cyberspace, digital forensics is used to investigate and analyze digital devices, such as computers, smartphones, tablets, or other electronic devices, to collect, preserve, and examine electronic data to establish facts or evidence. Currently, digital forensics is often successfully used in cases involving cybercrime, fraud, intellectual property theft, or crimes involving the location of personally assigned electronic devices like phones during a specific time. Digital forensic uses specialized tools and techniques to recover, analyze, and interpret digital data from various sources, such as hard drives, memory cards, cloud storage, or social media accounts. These data reconstruct events, establish behavior patterns, or identify parties involved in criminal or illicit activities.

Many forms of digital evidence are not accepted in court because they are easy to manipulate or falsify. But the assumption in digital forensics is that many data traces are left because it is not well known that they are being created in the background. Creating, modifying, or removing these digital data consistently is also very difficult. Even modifications or attempts to manipulate data could leave potentially suspicious evidence that would not be there if someone had not tried to cover up their involvement.

Depending on the assumed skills and access to tools, servers, or data, it is possible to narrow down the list of possible culprits. When it comes to state actors, they have assumingly access to advanced digital forensic labs and their tools to falsify evidence on a larger scale. It would therefore be a matter of trust to believe reported findings and the official narrative.

Still, humans, their organizations, or governments need to have some credibility and authority to lower the temperature, confusion, or excitement around enormously consequential events like a consequential Cyberwar 2.0. If no evidence is found on who is behind large (historical) events or how it has been done, this could create even more damage from anxiety or panic.

Additionally, giving human defenders in times of emergency a victory lap is good and essential for calming down a crisis because it could create some optimism that problems can be solved favorably without more drastic or disastrous means. Additionally, cyberdefenders and the public are unlikely to close the book on cyber capabilities without understanding how the hack is being done or what cyberweapons the winning party used. Having the case closed would also be in the interest of the operator of the offensive Hacker-AI. Their misdirection would be accepted as the mainstream explanation.

It is unknown how many exploitable vulnerabilities could help attack tools or malware entering systems, bootstrapping their capabilities, or elevating their rights or permissions. The easiest explanation is usually to blame humans for doing something wrong, like falling victim to a click-based vulnerability, which could easily explain some bad results. Days or weeks after malware was detected, users can easily be blamed for activating a link or confirming the installation of updated software that uses sysadmin rights. However, hacking systems' most dangerous methods do not need human involvement or confirmation. These attacks are called click-free.

There may have been attacks done years back, and the corresponding security holes were fixed long ago. But the attackers have left covert/private backdoors to be activated without human involvement. Click-free vulnerabilities can be used immediately without depending on users; that is why they are considered more valuable. Protecting these click-free exploits is a priority in which click-based vulnerabilities are potentially sacrificed.

Additionally, attackers should better not reveal that they were successful. Instead of stopping after entering a beachhead, they may leave evidence of a continuous flood of failing attacks. This could give cyberdefenders the impression that they were not successful yet. If there is no evidence of a successful entry, there is no cyberattack; the case must be closed sooner than later.

Less sophisticated actors usually do unsuccessful attempts. Their emergence could be part of a Cyber or Cyber Patsy. A patsy is made look guilty of something they did not do. He is used as a scapegoat or is set up to take the blame for some actions. It is conceivable that these patsies might be coerced to take the blame, or they are being tricked into doing so. Imagine, these patsies could play the role of a covert state-sponsored terrorist in which their criminal actions are denied, officially. Are they (really) associated with a certain nation, or was this an elaborate plan by some other intelligence service?

Cyber Patsies are cheaply blamed in the digital realm; they could be used to mislead investigations. They are used when there is evidence of an attack or if the public requires someone to blame due to the scale or damage from the attack. Hacker-AI-generated malware, Cyber Ghosts, or Devils could prevent output that would make humans suspicious. Also, creating patsies could be a deliberate strategy for blaming innocent parties when particular financial damages need to be explained. Some specific persons or groups must be responsible - following public sentiments and biases that could make this task easier.

Blame is often used to deflect attention away from the true perpetrators. Thereby, he avoids taking responsibility for the attack. This makes it more difficult to attribute cyberattacks accurately. An attack can be mistakenly attributed to the wrong group or entity, like a failed laboratory experiment with artificial intelligence (AI). With that kind of story, a unified effort could be created, and adversarial action could be taken to establish public safety. These events could be considered false flag operations in which attackers deliberately make it appear that the attack was carried out by someone else, like an AI. The main goal is to mislead or distract from adversaries' involvement and potentially create the necessary pretenses for additional steps that would not be acceptable under other circumstances.

In Cyberwar 2.0, we will assume that its main goal is regime change or overthrowing legitimate governments. Getting to that point may require some political theater. Designing the narrative around this drama is probably outside the technical aspects of Hacker-AI, but creating evidence for a Cyber Patsy. Example: The to-be-overthrown government could make its case to the public and blame the patsy; they still fail to keep in power. In that case, the old government and the blamed coup (patsy) government would fail, and the real instigators could present themselves as rescuer.

Preparing Cyber Patsies would start by sharing information accidentally with groups worldwide so that it appears as an active entity. Hacker-AI would be used to keep the story and the evidence consistent. Humans involved in this process would likely make mistakes - Hacker-AI could copy these kinds of mistakes without making a mistake. The goal of building a Cyber Patsies is to directly attribute cyberattacks or cyberwar actions to others so that the deflecting arguments and evidence would come from forensic experts independent of the real culprits.

(15) Attack Synchronization / Management

Hacker-AI will allow nation-states and anyone else developing it to use malware as cyberweapons more frequently and more targeted. Hacker-AI does not aim to kill people or destroy enemies' property or capabilities.

Sporadic use of malware from Hacker-AI would under-utilize its capability and keep kinetic or explosive weapons as the primary choice. Hacker-AI, in the hand of private organizations, could likely make their operators the new leaders of a country. The primary goal of Cyberwar 2.0 is government overthrow.

The speed, scale, and determination at which Hacker-AI could exercise coercive force on people within a cyberwar could make it a preferred weapon of choice in war - to trigger overthrowing legitimate governments / regime change. Until cyberdefense is not preventing malware from being the primary tool in cyberwar, Hacker-AI-based capabilities would constitute major tools. But they would need to be synchronized and managed in planning offensive military operations; this involves complex processes with many different steps and considerations.

From a high-level view on Hacker-AI and Cyberwar, planning and preparation is segmented into the following steps: definition of the mission, assessing situations, developing detailed plans, coordinating among stakeholders, and command / control over plan execution.

For defining the mission and objectives, Hacker-AI operators must identify targets and objectives that need to be achieved while outlining the actions that should be taken to support any broader strategic goal. These involved steps are outside Hacker-AI's skills. Additionally, the advantages of Hacker-AI could be quickly wasted. Using Private Backdoors or Cyber Devils, Hacker-AI could be leveraged to keep followers at a disadvantage in the long term. Short-term, Hacker-AI and Cyberwar 2.0 are probably most efficiently used in decapitating the leadership in a country or society via overthrowing the government and then establishing an AI-based surveillance system that would suppress dissent. A private business could also use it to control a country's government. Strategic goals will lead to more specific operational plans and the assessments of risks, potential collateral damage, or unintended consequences these goals could have.

Cyber Reconnaissance reveals which computer networks/systems or software vulnerabilities can be exploited and beneficially used. This surveillance determines weak spots of enemy civilians that need to be recruited by intimidation or bribes. Using gathering, aggregating, and analyzing intelligence, enemy's strengths, weaknesses, and likely tactics will be comprehensively assessed. It will be important to understand important factors that could affect possible operations. Hacker-AI could provide operational tools and reconnaissance details critical to successfully executing plans. Using these tools/data in realistic simulations, the assailant could predict likely outcomes and optimize operations toward the desired goals.

Hacker-AI in a Cyberwar 2.0 scenario supports primarily offensive actions; therefore, countermeasures of defenders will be taken into consideration, particularly how they could pose a threat to the cyberattack and later within the aftermath of a victorious outcome. E.g., what are the economic damages from possible sanctions? With data on projected consequences for the attacker's and the occupied country's economic shortages, they could start preparing.

Each move within a more detailed plan could be analyzed, prepared, and potentially scheduled or linked via trigger to other parts of the war plan. The attack is seen as a data operation; structuring and validating this plan is done by trained AI in simulations. Usually, complex operational plans are structured outcome-, function-, process-, or time-based. The advantage of AI is that it is much better at optimizing plans than the above approaches. It would likely not require following these methodologies when finding ways to put a full plan together.

The development of a detailed plan identifies and uses specific/prepared resources, develops a timeline for operations, and outlines contingencies. This step could also identify key milestones and decision points, including contingencies. Within this plan are automatic triggers for additional steps issued from the centralized attack management or decentralized hubs; humans cannot be trusted to keep up with these triggers. Feedback on progress must be designed to give operators a better understanding of problems or delays, but triggers will make the campaign continue. Also, policies and protocols must be designed to restrict or share information among the different parts of the operation. The cyberwar is a data operation in which the plan is realistically rehearsed via full simulations. These rehearsals will help involved units and personnel gets familiar with their roles and responsibilities.

Coordinating a cyberattack planned by AI will come down to having relevant parties know their roles and responsibilities. The attack management would have access to reliable and current information and resources. The fewer people involved or in charge of activities, the better this is for the plan. Where human judgment and decisions are required is difficult to determine. Cyberwar 2.0 does not involve heavy coordination with logistics/supply or communication among civilian organizations. After multiple simulations or rehearsals of the plan, confidence in the plan should be high enough so that the least risky method to operate the plan during the cyberwar operation is to listen to a recommendation by the attack synchronization management.

The war plan (or war script, as it will be called later) would be executed automatically according to timelines and objectives established in the planning process. Tabletop and in-field exercises have identified potential issues or vulnerabilities in advance. Key personnel review results and compares them with the simulated results. When all necessary resources and capabilities are in place, last minutes adjustments or plans that were not simulated in advance would make the outcome more likely worse.

Final Thoughts on Hacker-AI Details

No feature or Hacker AI detail is so difficult that it requires a state actor to get it done. On the contrary, if done in governmental programs, it may fail due to their lack of top-notch human talents who are more likely employed in private businesses.

If private organizations, without governmental blessings, would start developing these features, there is the risk that whistleblowers could reveal this, and the organization’s leadership could face serious legal trouble. As a non-lawyer, I am unsure which laws would be crossed for making only the preparation, not the operation, illegal. Some lawyers reading this may think that developing Hacker-AI, without deploying it or concretely preparing a Cyberwar 2.0, i.e., without waging a cyberwar or overthrowing a country’s government, could still be done legally. I don’t have a legal education, and therefore, no opinion on that.

However, the developments of Hacker-AI or Cyberwar 2.0 capabilities will not be done in public. It can be compartmentalized. Critical aspects can be disguised as research or even open source projects or sold to a company’s developer/engineering teams as national security, i.e., as part of a highly classified defense project.

I did not mention in the above features that some Hacker-AI features could have civilian applications. E.g., the tech library and tech simulator could be used to simplify the handling of legacy technology issues significantly. It will be a huge benefit for many if old products’ lifecycle is extended without further support, service, or maintenance risks. Other product features could be developed by having the attack parts disguised as the challenger component that helps companies to develop the corresponding countermeasures - i.e., training the cat by providing (good-enough) mice. Hacker-AI can also be disguised as a tool that helps cyberdefenders find and fix vulnerabilities.

Deep pockets combined with the technical leadership of people with vision and experience could pull off the development of Hacker-AI and other

Cyberwar 2.0 components in secrecy. Therefore, we may come to a scary conclusion: How many possible groups or individuals are around who could (theoretically) do that? And then, how many of them have a mindset that would allow them (actually) to start something like that for their own agenda? I doubt that the number is zero.

I assume that many filthy rich entrepreneurs live in exile for political or criminal reasons. They would love to use Cyberwar 2.0 capabilities to return to their country as (unchallengeable) political leaders. And then there are hundreds of filthy rich criminals or oligarchs who want to protect their freedom and wealth with some capabilities that anyone could ignore, even if they put justice over everything. Criminal state leaders could be brought to justice, with a handful of possible exceptions. But criminals who threaten the world with Hacker-AI could make themselves untouchable if they find the technical expertise to develop it.

Even if any of the mentioned suspects actively pursue Hacker-AI or Cyberwar 2.0 capabilities, they can be used as patsies. This might be how a superpower could utilize and normalize Cyberwar 2.0 as a potential weapon to occupy another country.

The problem for every superpower is that they are currently defined by the size of the military and their capability to wage conventional war. In a cyberwar, that’s irrelevant; the only thing that matters is (scalable) smart / Hacker-AI solutions. The amount of money or resources needed to become a cyber-superpower is minuscule compared to the other military spending. If we get to the point where we have a business like Cyberwar as a Service, with technology being used to change governments, then it is conceivable that there will be regions or countries where cybercriminals could be above the law. It is even conceivable that these people become even outside these countries untouchable.

The argument that no one would be so stupid as to create Hacker-AI or develop tools for Cyberwar 2.0 is wrong and irresponsible. Many stupid things are done, and the jails are full of people who thought they could get away with it.

However, I have a question: Do we want to be that vulnerable? Do we want to risk our freedom, safety, and security because we fail to solve a (solvable) technical problem? We use dangerously outdated cyber- and data-security paradigms; they don’t provide solutions. However, we need solutions urgently, or Cyberwar 2.0 could become a disruptive reality.