THE INTERNET, UNPLUGGED:

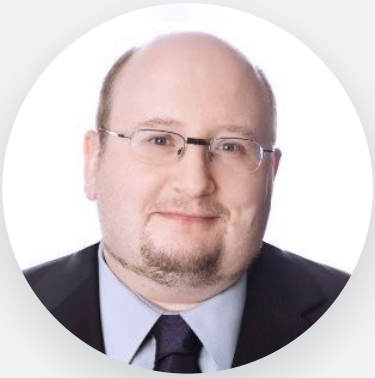
Your tech in a digital vacuum

Digital Security
Progress. Protected.

What if your internet came to a screeching standstill as a result of a natural or man-made disaster? How would you – and your computer running Windows and all manner of software – handle the disruption over a longer period of time? How would ESET's own software do in the same context?

*Read on as ESET Security Researcher **Cameron Camp** sits down with ESET Distinguished Researcher and Windows expert **Aryeh Goretsky** to answer these and related questions.*

Cameron Camp



Aryeh Goretsky

So, let's say a disaster strikes and you find yourself in a 'digital vacuum'. What are the first signs you'd see in functionality degradation, and how soon would those begin to be a factor?

If you have ever performed a clean installation of Windows, one of the first things you would notice the operating system does after installation is warn you that it is missing updates that affect its quality and security, and prompt you to begin the process of downloading and installing these. So, by default, you should understand that when you are performing a clean installation of the operating system, you are starting with some functionality degradation because of all the as yet uninstalled updates that have been released since the installation media was created. Windows has a couple of other functional considerations, too: If the copy of Windows is not yet activated, you will be prompted to d o so. In Windows 11 Home it is now a requirement to either create or log in with an online Microsoft Account as part of setup, or the process will not continue. This will cause problems if you are trying to install (or reinstall) that operating system in a digital vacuum.

Even if you are able to install Windows without having to create a Microsoft Account, the operating system must still be activated. If the computer on which Windows is running has digital entitlement, activation will occur in the background (its hardware has been associated with a product ID code used to activate Windows). If there is no such entitlement, the product ID code has to be entered. Again, this requires internet access.

On Windows 10 and 11, failing to activate Windows means the Windows Desktop will be watermarked or "tattooed" with a message stating that Windows has not been activated. Also, Windows' personalization features will be disabled. While this may not pose any security risks, it may be an irritation.

Recent versions of Windows periodically have to check in with Microsoft's activation servers in order to "re-arm" their Windows activation, or else they can get back into an unactivated state. The checking is done silently, so most Windows users never notice it. Historically, consumer editions have been able to run for six months without being able to check in. For enterprise versions of Windows, it can depend upon the types of licensing and methods used to activate Windows.

Same question security-wise on the Windows platform?

Now we're getting into some more interesting territory: Windows is one of the most widely used operating systems, which means it is one of the most widely attacked operating systems. As I mentioned previously, one of the first things Windows tries to do when it is installed is update itself. And those update checks continue running through the life of the operating system installation.

IT professionals have largely become used to Patch Tuesday, the second Tuesday of the month when security and reliability updates are released for Windows – and Patch Thursday, the fourth Thursday of the month when various non-security updates may be released. Not counting any emergency "out-of-band" updates, security updates occur monthly. The moment they are released, sophisticated and determined adversaries will begin reverse engineering those security updates to determine what vulnerabilities they fix, and then create exploits to attack unpatched machines. How long that can take varies on a case-by-case basis, but it could be weeks or days before the first exploit appears as a proof-of-concept. However, it also can be a matter of hours.

There is always a gap between the release of security updates and them being applied, especially in an organization where they may need to be tested to ensure they do not break any functionality with important line-of-business applications. That gap between the time the patches are released by Microsoft, and they are applied by Microsoft's customers represents a kind of "golden hour" for an attacker. That is, assuming they can get past all the protective layers of the customer's security to utilize the exploit.

The customer's in-depth defenses might block the exploitation of a vulnerability simply by preventing an attacker from accessing their systems. Conversely, many attacks are opportunistic: The attackers have performed some simple scripting to automate their attack, trying to connect to one public IP address after another. The attackers do not even need to be present or may not even check the results until after the scan is done. At some point, this programmatic scanning is going to get lucky and find vulnerable systems somewhere. The attackers can then look through their logs at their leisure to determine which systems to attack.

Are older platforms safer, since they weren't cloud native when built?

What does "cloud native" mean? In 1995, when Microsoft introduced Windows 95, one of its signature features was that it included a TCP/IP protocol networking stack. Microsoft even boasted that all versions of Windows after that would come with TCP/IP. But even prior to that, various TCP/IP stacks were available, both from Microsoft and from a variety of third parties (NetManage Chameleon, Trumpet Winsock, WRQ Reflection, and so forth) ...

... with varying features, price points, and stability. There were also TCP/IP stacks that could be purchased for DOS and OS/2.

The point is, network connectivity is **everywhere**: these days it is not so much a question of whether a local, metropolitan, or wide area network is connected to the internet, but how and where the connections occur, and what bandwidth is available. Sure, air-gapped networks exist, but at some point, they may be bridged with another network, or an external device may be connected to them (accidentally or otherwise).

With the understanding that even old versions of the Windows operating system have had network connectivity since the 1990s, we can try to distinguish between those which were designed before network connectivity was pervasive and "always on" and those that were not, but even that becomes somewhat fuzzy territory. What is the point that demarks pre-cloud and cloud-native operating systems? The ability for the operating system to automatically download updates from the internet without having to use a web browser? The ability to store files in the cloud using Apple iCloud, Google Drive, Microsoft OneDrive or other cloud storage provider's service?

Maybe the point of delineation isn't the operating systems that were used, but the speed of their network connections? Microsoft Windows XP received updates and support from 2001 to 2014. During that fourteen-year period, many of you likely went from dial-up networking speeds of no greater than 56Kbps to broadband connections in the tens to hundreds of megabits per second at home, while in corporate networks Gigabit Ethernet expanded from the network core out to the periphery of where the desktops and laptops were located. And those connections were great for network-transmitted worms, which brings up another question of where the delineation point might be: Could it be the defining point is where the criminal ecosystem caught up with the networking ecosystem to the point where network-centric malware could successfully spread? By the early 2000s, there were plenty of worms that spread via email, internal network shares, or the public internet.

Now, it may be possible for a small or even a medium-sized business to remain disconnected simply by virtue of having no actual physical connection to the internet, but as an organization grows, so does "shadow IT", and that means having to deal with employees who may bypass IT department restrictions and implement their own internet access through cellular hotspots. And if internet access is restricted at the regional or country level, users at the border may still be able to connect to cellular service from providers across the border, while satellite access may be an option countrywide.

The point is, the idea of an invulnerable platform is something of a myth: The longer a platform has been out of support by its developers, the more unpatched vulnerabilities it is going to have to exploit. So, the choice becomes believing you can rely on an older platform hoping that security-by-obscurity protects it, or using currently supported operating systems and applications, and keeping them patched against security vulnerabilities.

I will note that there are security companies who do continue to provide support for their products on operating systems that are no longer maintained. ESET has historically done so, ...

... and is not unique in this regard. While these can block threats, they cannot remediate the underlying vulnerabilities in the operating system. For that to occur, the vulnerability must be removed via an update that patches the code that allowed it in the first place, and that code resides in the operating system. Or occasionally you might get lucky and a vulnerability can be defanged through a configuration change, and the required change does not interrupt some crucial functionality you depend on.

How would ESET software do in the same context?

Well, ESET has many different programs, and each of them has several versions that are supported, but for the point of this discussion, let's talk about the current versions of our endpoint protection programs, which run on desktop and server operating systems. Before I begin to answer that question, though, I think it is important to understand how malicious code is identified. This entails a rather long and winding discussion, for which I apologize in advance.

In his 1986 doctoral dissertation, Fred Cohen formally defined what a computer virus is, and identified three ways to detect viruses:

1. By computing a mathematical value for known-good files such as a CRC or hash and checking files to see if the precomputed mathematical value for them had changed, which would indicate the file has been altered or tampered with in some way. This forms the basis for allowlisting, denylisting, blocklisting, and related mechanisms used to authenticate a program.

2. By identifying specific pieces of program code that have been explicitly determined to be malicious. This could be through matching a specific sequence (or sequences) of bytes or API calls, hashes, or similarity matrices. Historically, this was referred to as fingerprinting or scanning, but such terms are no longer regularly used in the industry.

3. By identifying an action (or set of actions) being performed by the malicious code, preventing it, and sometimes alerting the user when the attempt is made to perform these actions. This is known as behavioral detection, since it relies on identifying a type of activity, rather than focusing on specific sequences of code.

In the earliest days of computer viruses, the first antivirus programs typically focused on using one of these methods and that method only, but by the late 1980s and early 1990s, antivirus programs began to combine and build upon these various techniques. Today, all detection mechanisms rely on some or all of these three "atomic" mechanisms as ...

... building blocks. Even the most advanced machine learning and neural network algorithms use them in some fashion to determine whether program code is a threat, is not a threat, to assign the probability of a threat, and so forth.

This brings us to the next part of the discussion, which is somewhat philosophical in nature. When it comes to designing and architecting security software, which of these mechanisms do you choose to use and build upon to develop your own detection mechanisms? Once you have made that decision, you next have to decide in what ratios they will be used, and in what contexts do you apply them, as not every mechanism may be appropriate (or even useful) in all situations. Over the last decade, another question to be asked is what level or where do you apply them? Every detection technology has a cost associated with implementing it, whether that's in processing time, disk or memory I/O, memory allocation, disk space, and so forth. Moving some of that into the cloud can have benefits to protection, but there are some tradeoffs as well.

So, with all of that in mind, what does it all mean? It means that it is important to choose the right means of detection for each threat: otherwise, your detection engine is going to perform poorly over time as new threats are added to it.

ESET's founders released their first antivirus program not too long after the first DOS computer viruses appeared, and in the intervening decades has gone through multiple cycles of innovation, adding new technologies and protective layers to its software and so forth. But those have always been based on the previously mentioned atomics, and just because a new detection technology has been added does not mean a previous one has been removed. In many cases, an existing detection method may be the optimal one for a certain class of threats. In fact, a new detection technology may be useful for detecting, say, some new exotic class of threats used by nation-state threat actors in edge-case scenarios, but perform poorly against conventional threats seen on a daily basis.

It is extremely rare, at least in ESET's case, to remove a detection mechanism in its entirety. The only instance I can think of where this occurred in recent memory was when we upgraded our signature-based detections with DNA detections, a considerably more efficient and effective method of detecting harmful software. That was the end result of a specific set of decisions to create a technology to replace signature-based scanning, and it turned out to be the correct set of decisions. That said, all of the tooling for signature detection is still available in case a threat appears that explicitly requires it, but the DNA detections have been highly effective.

Earlier, I mentioned that there are different layers where detection is performed. The advent of cloud computing has allowed many security vendors to move detection processing from local computers (which can have processing constraints) into the cloud, where dedicated computation can be performed continuously for threats. Some vendors perform most of their threat detection in the cloud; this has the advantage of allowing them to perform analyses that could not be done due to performance concerns locally, but that also can result in delays or missed detections due to unstable internet connectivity.

ESET has moved some of its detection layers into the cloud, but on a selective basis. Examples of this include LiveGrid, which consists of several interrelated systems used for blocking both files and URLs, the submission of previously unseen and potentially malicious code, and the reputation of objects; and LiveGuard, which is ESET's cloud-based analysis platform for unknown and suspicious files. Cloud-based detection, reputation, and analysis technologies are not unique to ESET, nor are they particularly new technology: for decades, security software developers have engineered their own automated malware sample submission and analysis platforms, but they largely worked in one direction only, from endpoints into servers where suspicious files would be analyzed, and signatures created that eventually reach the endpoints in the form of updates. This pipeline fed into other systems to provide telemetry, prevalence, and other metadata useful for protecting against threats. The gradual change was in more and more of this data being fed back to endpoints after automated analysis, instead of remaining and being used solely in-house.

The LiveGrid and LiveGuard cloud-based systems developed by ESET have followed this path into the cloud. Again, this is not something unique to ESET; other security vendors have done the same with their backends. What is different, though, is the approach ESET takes to what it does in its cloud. I have already written about some of these technologies so I do not want to repeat myself but will note that they occur at a far larger scale than what can be done on an endpoint without imposing significant costs in terms of performance and usability.

In ESET's case, detections are often first published to the cloud before being distributed in the form of downloaded updates. The speed at which the cloud can be updated and allow lookups means that detections can be published there first, especially high priority ones that cannot wait until the next scheduled DNA update. It also allows detections to be withdrawn more quickly in the event of a problem.

I can talk a lot more about the various detection mechanisms used by ESET; however, this is not the point of this article, so I'll conclude this answer by mentioning a detection mechanism that is very relevant to this discussion: not only does ESET utilize multiple layers of detection technologies; in some cases, some detection technologies are literally built from others. For example, certain types of ESET's generic detections for never-before-seen malware are automatically generated. This is based on the data it stores to detect existing forms of malware. What this means is that each time the program receives an update to detect existing, known malware, its ability to detect malware not yet seen by ESET improves as well. This type of protection is all performed locally, so it occurs even when there is no internet access to the cloud.

With that discussion concluded for now, I would like to talk a bit about ESET's connectivity requirements. As mentioned previously, LiveGrid and LiveGuard only work when there is an internet connection, but these can be disabled if no such connection is available. If there is no connection to the internet, though, this could not just impair the ability to detect new malware, but to also remediate false positive detections, and use of LiveGrid to determine if new software is prevalent, trustworthy, and so forth. But even the detections that are stored locally have to be updated at some point in order to protect the computers and networks on which they are running, and that means having access to ...

... the public internet to download those. While ESET's business offerings can be updated from repositories on an internal network, or even from a USB flash drive in the case of an air-gapped network, at some point, the detection update would have to have been updated from ESET's public-facing update servers on the internet. There is even a provision for updates over very limited satellite connections, which could be an option to get around a digital vacuum. ESET's software will continue to run – and provide protection – indefinitely, but after seven days without downloading updates, the user interface will display a warning that the program may be out of date.

What technology should you look into now, if you anticipate going digitally dark?

From a physical point of view, being able to maintain your internal or domestic networks means you are going to need a supply of parts to keep those networks, and the computers attached to them, functioning. Most consumer computer and networking gear these days do not ship with schematics and are not designed to be user repairable at the component level, so it becomes a necessity to keep shelf spares, i.e., entire swappable devices, in inventory to replace those that stop working. A supply would also need to be kept for replaceable components that can be swapped out, such as fans, drives, expansion cards and so forth. This would allow IT to keep equipment runnintg until normal internet connectivity was reestablished. In 2018, I wrote a blogpost and accompanying comprehensive white paper, The Last Windows XP Security White Paper. A section of the paper specifically dealt with how to maintain the hardware used to run that old operating system and its software. Many of those concepts can be applied to current versions of Windows or even other operating systems.

For threat protection, it would be important that the security software be capable of functioning without an active internet connection. While having cloud-based detections enhances threat protection (as noted above), whatever you use should still be capable of providing a high level of protection against known and unknown threats when that connection is unavailable.

Another consideration is that without access to the internet, all backups will be local, or even regional at best, as even offsite backups will likely be to storage located in the same region. The ability to not just store that information, but also maintain the systems responsible for backing up and restoring data, will become critical infrastructure. Without the ability to access the global internet for storing or retrieving data, only local information will be available, and it will all be backed up to and restored from local resources. Back in 2011, I wrote a paper on backup basics, intended for home and SOHO users to familiarize themselves with various options for local backups and help them find a method that worked for them.

While its discussions about storage capacities and pricing are now long out of date, the underlying concepts are sound and may still provide some education, although they may not scale upwards very well for enterprise-level requirements.

If the outage were to continue for an extended period, the overall storage capacity of a backup system would gradually decrease as media is consumed, drives fail, and so forth. So, techniques like file deduplication would be critical, and decisions would have to be made about what information to continue backing up, and what not to keep backing up. And, of course, backups are only useful if they can be restored, so any long-term plans for backing up information should also include testing restoration capabilities.

Another consideration would be the encryption of information. While it is standard in some organizations to encrypt storage as well as data communications, it is not universally required. Depending upon the reasons for going digitally dark, there may be a need to encrypt drives using disk encryption software, as well as secure network connections through an encrypted VPN tunnel, IPsec, etc.

In some circumstances, it may be necessary for some equipment to have a physical kill switch to perform a wipe of information stored on it, as remote wipe capabilities would be unavailable due to the lack of internet connectivity. That is, of course, entirely depending upon the reasons for going dark, and whether the need is to perform a device wipe or to physically destroy the hardware.

What about physical safety: if you were in a situation where you suddenly had to grab your tech and go, how would that change your calculus?

This is actually a scenario I had to deal with as an individual a while back due to a wildfire. In this instance, the tech items that were grabbed were laptops and external USB drives used for backups, which went into the car. Desktops, servers, monitors, networking gear and similar kit were powered down, unplugged, and left in place. While replacing all of that hardware would be both time-consuming and expensive, it is all replaceable. The irreplaceable part is the data, and that is what you want to focus on taking with you if there is a situation where you need to leave. If you are facing a natural disaster and have some time to prepare, taking a video of your belongings on your smartphone and narrating what they are can be helpful when dealing with the insurance company.

There is quite a difference between evacuating because of a natural disaster and evacuating because of hostilities in the region. In my case, all of the storage was encrypted and protected with unique, strong passphrases. While this would not have prevented the loss of my ...

... data due to the hardware being damaged, stolen, or lost, it is also unlikely the encrypted data would be accessed by a third party. It is an entirely different scenario when fleeing due to a conflict and the authorities might take a great interest in your data, and even compel you to decrypt it. In a situation like that, hiding the hardware storing your data may be just as important as ensuring it is encrypted. You may even want to have decoy devices to show the authorities but remember they should have some innocuous data on them. A device without address books, dialed phone numbers, saved pictures and videos, social media accounts, bookmarked web pages, and other signs of use is inherently suspicious and may trigger unwanted attention.

Companies are going to cloud-only or subscription-only versions of software they don't own. In the future, how will they be able to deal with this if the cloud gets disconnected?

Some cloud-only software and services are entirely web-based, which means that without access to the internet, they are not going to work, or at least not work very well. If software (or service) is hosted internally or in a regional data center, then it may continue to operate normally for a while. However, even if the data center remains operational, there could still be issues due to problems with network routing, domain name servers going offline, or portions of the physical network being damaged or destroyed.

Subscription-based software may periodically check in with licensing servers, or for specific functions, such as being moved to a different computer, changing the number of seats licensed, increasing the performance of the hosting server(s), and so forth. If the software cannot do that, it may revert to an unlicensed mode with reduced functionality, or stop working altogether. There may be a desire to look for patches or cracks to bypass these issues assuming there is some internet connectivity, but those can be a source of malware such as information stealers and ransomware. That then becomes a risky activity, especially if the security software no longer has access to its cloud for analysis.

We see nation states taking a more active role with targeted exploits against a particular geographically determined adversary. From the defender's perspective, is there any way to specifically block attacks of this type?

This is a very difficult question to answer, because (1) there is a lot of nuance as to what can be considered an attack; (2) who should be responsible for protecting against them; and (3) how the attacks are blocked. The use of malware for military objectives is different and occurs outside of the "norm" of criminal activity, where the end goal is financial in nature. While there may be similarities in terms of responses and countermeasures to criminally motivated malware that can be applied to nation-state malware, there are some concepts, such as retaliation, that are simply out-of-scope.

There are many aspects to defending against targeted attacks, and time, budget, resources and scale all factor into that.

- If you are using commodity hardware and software, there is probably information about hardening them available from various public and private sources. You may even be setting up an agency to perform this activity, both by studying existing work in the field as well as performing original research. These would then need to be implemented as mandatory requirements for the configuration and deployment of computing gear.

- You may develop your own hardware and software for national use, as China and Russia are trying to do. Few economies operate at a scale to allow this, and given the complexities involved it could be years before they are successful. Having a completely closed computer ecosystem would make it difficult for an outsider to attack if the architecture is completely unknown to them. Unlike in the movie *Independence Day*, it can take weeks or months to develop offensive computer attack capabilities, and that is when the attackers already have years of experience with their victims' operating systems.

- A different solution might be to look at vintage hardware designs and concepts, and see how these can be applied to modern technology. To be clear, I am not talking about using actual hardware and software from the beginning of the microcomputer era, but rather take ideas about how they operated.

Many of the first personal computers had their operating systems and applications stored in mask ROM or EPROM chips. In some cases, the applications were built into the computer; in other cases, they were sold as plugin cartridges. Unlike modern EEPROM chips, which can be reprogrammed (aka flashed) with new software, ROM chips cannot be reprogrammed at all. EPROM chips can only be updated if they are removed and installed into specialized hardware called an EPROM burner for them to be reprogrammed. For these types of chips, updating them means removing and then replacing them with new ones. To be clear, this does **not** mean that these types of chips are attack-proof: A supply chain attack or a manufacturing attack could result in malicious code being implanted, and the processes through which the updates are distributed and applied will be targeted as well. However, physical access becomes more difficult as controls are put into place to prevent it.

Modern operating systems and applications may periodically need to write to the drive they are running from, whether for temporary storage or to save data files. They may also need to be updated for legitimate reasons, from time to time. The need for updating may preclude ...

... ROM-based software. An EEPROM – which can be reprogrammed in situ – might be a solution if it is coupled with a physical write-protect mechanism. Early hardware containing EEPROMs often had switches or jumpers that needed to be shorted in order to allow them to be rewritten. This feature eventually disappeared from commercial hardware to cut costs, but perhaps it would make sense for the defender to reintroduce it in systems where the ability to survive an attack takes precedence over manufacturing costs.

In one of my previous answers, I talked about the need for backups. One problem with backups is that they are a target, too. An overwritten or damaged backup may be immediately recognizable; one with data subtly altered in it, less so. For the defender some form of Write Once Read Many (WORM) media for data storage may be desirable. As the name implies, WORM media can only be written to once, but can be read over and over again. An early, non-digital example of this technology is phonograph records: They are manufactured with the music stamped into them: You can listen to them all you like but cannot record new music to them. A final point I will make about backups is that they need to be stored somewhere, and that location may itself be subject to a kinetic attack.

While my comments here have largely been hypothetical, I will point out there is some very real information about actual attacks and defenses to them in the ESET Ukrainian Crisis Response Center.

Thank you, Aryeh, for all the tips and advice. We hope this will serve as a multipurpose resource for those needing or wanting to disconnect, but still assure reasonable levels of protection. We may revisit the subject in the future as the variables change, but being prepared seems like a perennially good idea, internet connection or not.

Thank you, Cameron, for a most interesting discussion.

Digital Security
Progress. Protected.