A WEBGAP WHITEPAPER ON

Browser Isolation Cybersecurity

The Concept, Different Models & Their Application

WEBGAP INC

340 S. Lemon Ave.
Walnut, California
CA 91789, USA

(415) 520-3570
info@webgap.io
https://webgap.io

Table Of Contents

Introduction

Browser isolation is a modern reinterpretation of the old security through physical isolation model, also known as air-gapping for those of you who remember the term.

***An air gap** or **air gapping** is a network security measure employed on one or more computers to ensure that secure IT infrastructure is physically isolated from the public internet.*

In the early days of computing it was quite common to see people sat at their desks in high security environments with two or more computers in front of them, each computer connected to a network that was physically isolated from the other. Their users needed resources from those computers to do their work, but it was too much of a risk to have those resources all on one machine that was connected to the internet.

This was done to ensure that any hackers who managed to infiltrate from the public internet could not then break into an adjacent computer on another network. Back then hackers could not jump through thin air and it was upon this notion that the concept of air-gap security rested. Those who did need access to the internet simply got a second internet connected computer on their desk to browse the internet on.

The browser isolation cybersecurity model takes this concept of security through physical isolation and applies it to the modern day problem of malware and cyber attacks against internet browsing users. Because the vast majority of attacks directly target users through their browsers, it makes perfect sense to isolate their browsers and the associated cyber risks away from your the rest of your internal IT infrastructure.

Isolating your organizations browsing activity and the associated risks onto a platform built to handle that risk is the most effective way to improve your cybersecurity posture over the long term. It enables you to close down the most common infiltration points on your networks, freeing up your resources to focus on securing the rest of your IT.

With Gartner predicting that more than half of US businesses will begin to isolate their browser over the next three years, representing more than 25 million businesses and more than one hundred million internet users, it is important that we take a closer look at browser isolation cybersecurity so that we may better understand it.

In this white paper we will take a closer look at the concept of browser isolation cybersecurity, it's different models, and their application in the real world.

Browser Isolation As A Concept

Browser isolation as a concept emerged from the idea of ***security through physical isolation*** and the security practices of cybersecurity professionals who have long been physically isolating workloads into separate physical or virtual machines.

In retrospect browser isolation as a concept seems perfectly intuitive once it is explained to you, because we have always been isolating risk one way or another. Take for example handling nuclear materials, long ago scientists developed a radiation 'glovebox' to safely isolate the risks of radioactive contamination when working with radioactive materials in their experiments, it keeps them safe from the associated risks.

A glovebox in a nuclear laboratory is a sealed container with plastic gloves that you put your hands into in order to touch what is inside the box, it allows a person to work with hazardous substances and not contaminate themselves or the laboratory. When you think about it, this is exactly what browser isolation hopes to achieve but instead of radioactive materials in the glovebox it is the internet that we wish to touch.

Browser isolation as a model began at Los Alamos National Laboratory (LANL.gov) and Lawrence Livermore National Laboratory (LLNL.gov) and because were NNSA nuclear laboratories they thought of the model as ***an internet glovebo***x. The first commercial browser isolation platform was developed and deployed at Lawrence Livermore National Laboratory (LLNL.gov) back in 2010 and I was privileged enough to lead the team that designed and deployed this solution, we called it SafeWeb back then.

These projects are considered to be the birth of the modern browser isolation space and back then there were no remote browser vendors, the industry did not exist and nobody understood what you meant when you said 'remote browser' or 'browser isolation'.

Browser isolation model was born out of necessity at these facilities, highly secret NNSA laboratories with thousands of federal employees using the internet and constantly under malware and cyber attack. They understood that they needed to physically isolate their users web browsing away from their internal infrastructure in order to secure their networks and enhance their overall cybersecurity posture, they knew that they needed to build an internet glovebox in order to properly isolate their organizations cyber risks.

The Evolution Of Browser Isolation Technology

Looking back it seems intuitive to physically isolate cyber risks, but at the time nobody was doing it and we did not have the right tools to deliver the model in an efficient or scalable way. Because browser isolation was still in its infancy, the only tool we had for the job was virtualization, a technology designed to consolidate server workloads pressed into a different kind of service isolating browser compute loads.

Around that time (2009) virtualization was spreading from servers to the desktop and seemed to be the best tool for the job. The early implementations of the browser isolation model leveraged desktop virtualization which was hugely inefficient and expensive. It was expensive because there has never been a point when SAN centralized virtualization could be considered cost-effective at any scale and also because instead of just virtualizing the browser in order to isolate it, we had to virtualize the whole desktop OS and its underlying hardware resources.

Just like in the old days when each user was given a separate computer connected to a separate network, users were given a separate hosted virtual desktop upon which they could browse the internet safely from outside their internal networks. Despite getting the isolation job done, this was complete overkill. Desktop virtualization was never a fit vehicle for handling the browser compute load, it multiplied the costs far beyond what they should have been when you take into account the browser workload.

We just did not have a specific technology that allowed us to effectively isolate the web browser and so instead of isolating a small parcel of resource (the browser compute load) into a small box, we isolated a large amount of resource (the desktop OS compute load) into a very large box and thought we solved the problem.

This is the real reason why browser isolation has not been adopted on a mass scale, despite it very clearly being a highly effective endpoint security solution, the underlying tools we use to isolate the browser are still inefficient and expensive. The browser isolation space has never really grown out of using virtualization as a tool of browser isolation which is a roadblock to the mass market adoption of its solutions.

If you took one look around the browser isolation space today, you will see that this problem has never really gone away over the last five years, instead vendors are amplifying the problem by doubling down on the non-persistent virtualization model.

Many of the leading vendors in the market (Menlo Security, FireGlass and Authentic8) all leverage the same legacy SAN centralized, virtualized architectures and bring nothing new to the table in terms of innovation around handling the browser compute load. It is for this reason that browser isolation cybersecurity has still not been adopted by the mainstream despite the obvious benefits. It's just too expensive a solution for anyone other than large organizations with dedicated cybersecurity budgets.

This is why browser isolation technology remains a cybersecurity solution for the few rather than the many and this is a real problem over the long term. When a technology proves itself to be highly effective at stopping cyber attacks but it is priced too high for those who really need it, then the technology is really not fit for market purpose.

It's not fit for purpose because the whole point of browser isolation is to protect the many rather than just the few and the browser isolation industry is failing to do that.

When you look at browser isolation technology through this lens, it becomes immediately obvious that the legacy models are not meeting the long term market need even though their vendors might be making money over the short term satisfying the security requirements of those who can afford their legacy isolation solutions.

It is clear to almost everyone except for the shareholders in these vendors that SAN centralized, virtualization based architectures are not the future of browser isolation mass adoption. It is obvious that technologies which directly address the core problems in browser isolation need to be developed if we are ever to approach mass adoption and protect the many rather than the few against web based cyber attacks.

Shoehorning old models into browser isolation does not really solve any of the real long term market problems. My team and I learned these lessons the hard way almost ten years ago hosting browser isolation platforms for thousands of federal government users. We understood back then that the key to mass adoption was to dramatically lower the costs and exponentially increase the scalability of remote browser platforms.

There are 300 million internet users in the US alone, so a platform that is 'fit for purpose' has to be able to cost-effectively accommodate at least one million simultaneous users and virtualization based solutions cannot ever hope to get us there.

The Real Problems In Remote Browser Isolation

Anyone can properly isolate a browser or a desktop, but the trick to unlocking mainstream adoption is being able to do so in a cost-effective and scalable way.

The real problems in remote browser isolation are cost and scale and they have nothing to do with isolating the browser effectively. If the only requirement was 'isolate the browser' then virtualization is the perfect tool for the job, but in the real world we have to take financial budgets and the actual number of users in the market into account.

Cost Is A Problem

According to a Desktop TCO study from Gartner, the current endpoint security spend for the average organization is approximately $56 per user **per year**. That is all the average organization in the US has in their budgets to spend on end user security.

When you consider that the current market cost of a remote browser account ranges $25-$75+ per user **per month**, you realize that existing browser isolation solutions are just far too expensive for anyone in the mainstream or small business owners to seriously consider as a financially viable cybersecurity solution. That's a problem because every internet user in the market needs a remote browser, but if they cannot afford one they will continue to suffer from web based cyber attacks.

Scale Is A Problem

Let's talk about the market for a moment because it is far bigger than most of us can comprehend and in order to keep the numbers low I am going to focus on the US browser isolation market rather than a global one. According to the Pew Research Study On American Internet Access there are 287 million internet users in the US and 124 million of them use the internet while sat at their computers on a daily basis.

From a vendor perspective those numbers are mind-bogglingly big, but obviously no one vendor is going to be able to single handedly capture the whole market, so let's break these numbers down a little in the serviceable available market (SAM). According to Gartner 20% of US organizations will deploy a remote browsing solution over the next three years so that's approximately 20 million users we have to consider.

Assuming that a vendor wants to capture 10% of that market, their platform needs to be able to accommodate at least 2 million users and right now none of the vendors operating SAN centralized, virtualization based platforms can hope to meet that need in anything like a cost-effective way, even when you factor in the efficiencies of scale.

Even the largest organizations would struggle to scale a SAN centralized, virtualization based platform for two million users, it's such a mind bogglingly big infrastructure project that nobody would every attempt it and vendors have not had to scale that high yet because the remote browser market is still so young. The biggest deployments you see out there still run into the thousands of users rather millions of users.

When vendors can build lots of private virtual clouds for small amounts of users the problems of scale do not readily present themselves, but when you consider the size of the market you realize the problems with scale are never going to go away.

Difference Between Browser Isolation & Remote Browsing

It is worth touching on the difference between browser isolation and remote browsing briefly, if only because some vendors insist on adding the word 'remote' onto 'browser isolation' and calling everything that our industry does ***remote browser isolation***.

Remote browsing is the practical application of browser isolation technology, as seen from the perspective of the end user who uses a remote browser.

Remote browsers are hosted on servers and then delivered to the user as a service over the internet, the underlying technology used is browser isolation technology.

I think it is important to clarify what remote browsers are, because the server-side computing model of browser isolation that they are based on is not the only model. We also have client-side browser isolation solutions in the market too which could never be described as 'remote browsing' because everything is executed locally.

To refer to browser isolation as 'remote browser isolation' is an attempt to confuse the market into thinking that there is only one kind of browser isolation model and completely ignores the different kinds of browser isolation models that exist.

Server-Side Browser Isolation Models

When discussing browser isolation it is helpful to distinguish between the server-side and client-side browser isolation models. Server-side models deliver a remote browser to their users, one hosted on a physically isolated server built to handle the cyber risks.

The primary advantage of server-side browser isolation models is that they stay faithful to the ***security through physical isolation*** model and quite literally physically isolate malware and cyber attacks away from your networks and users machines. That real physical isolation is the most important part of the browser isolation model.

When it comes to web-based malware attacks against secure IT systems, we trust the air-gap and always have done. Ignoring some of the some very rare and sophisticated out-of-band attacks designed to breach the air-gap, the most effective way to protect a group of users from malware attacks is to physically isolate them from the internet.

There are different kinds of technologies used in server-side browser isolation models and these can be split into two types; virtualization based or containerization based.

Server-Side: Virtualization Based Browser Isolation

Some of the leading vendors in the browser isolation space leverage SAN centralized virtualization based platforms. The less sophisticated vendors are using server virtualization and the more sophisticated vendors are using non-persistent VDI.

The problems you discover using virtualization as a tool of browser isolation are exactly the same problems you find when using desktop virtualization in general, it is hugely expensive and requires lots of servers and SAN to scale in any significant way.

It used to be a running joke in the desktop virtualization space that the only people who were actually making any money were the SAN vendors and this is still true. If a 10,000 user VDI platform is an expensive and infrastructure-heavy beast, imagine the expense and hardware overhead of a 500k user browser isolation platform based on VDI.

Each virtual machine runs not just a full copy of an operating system, but a virtual copy of all the hardware that the operating system needs to run and this quickly adds up to a lot of RAM and CPU cycles for browser isolation workloads, it’s complete overkill.

Also consider that virtualization based solutions must rely on the same display presentation protocols that VDI platforms rely on, meaning that these solutions cannot ever properly integrate with your local browsers for a real native experience, and by extension never be compatible with the browser plugins that users want to use.

Ultimately virtualization is a legacy hangover from the days when it was the best tool that we had to effectively isolate a users browsing activity. When it comes to making long-term browser isolation purchase decisions it is wise to take heed of the federal government consensus that it is too early to commit to any specific vendor in a significant way because the browser isolation space is evolving so rapidly. This represents an accurate, if brief, assessment of our space, it's new and still trying to work out the best model of triggering browser isolation mass adoption.

Server-Side: Containerization Based Browser Isolation

To avoid these problems my team and I at WEBGAP moved away from SAN centralized virtualization as a tool of browser isolation and towards distributed containerization, which we thought was a much more efficient vehicle for handling browser loads.

With containers (instead of virtualizing the underlying computer into a virtual machine as you do in virtualization) just the OS is virtualized, making containers exceptionally light, small in size and lightning fast on start up. All that a container needs is enough of an operating system to run a specific program, which means that you can put 5-10 times as many isolated browsers onto a server with containers than you can with virtual machines.

My team and I realized early on that containers are a much more efficient vehicle for handling the browser compute load, because **container based platforms require approximately ten times less server infrastructure** than comparable virtualization based browser isolation solutions, representing a significant drop in overall costs.

When it comes to handling something as small as the browser compute load cost-effectively at a very large scale, containerization is a natural vehicle.

The advantages of a distributed architecture compared to a SAN centralized architecture are already well documented, but distributed architectures offer some additional advantages when it comes to isolating browsers at scale.

There is no single point of failure and no centralization of resources which could lead to bottlenecks and scalability problems. This is important in browser isolation because if an intruder manages to breach part of your infrastructure, you can quickly shut down the underlying resources to close out the attacker and continue to deliver a service using the remaining resources that have been distributed across the network.

With a distributed architecture browser isolation instances and their supporting hardware are distributed across the network which gives a network distributed browser isolation platform an incredible amount of resilience and massive scalability.

It is the network distribution which makes platforms so massively scalable and the only real limitation of a distributed browser isolation platform is the size of the network. It is also a good way of delivering a low-latency, high quality of service, delivering services to users from the nearest geographical node on the network. Right now the only vendor with a containerization based browser isolation platform on the market is WEBGAP.

Client-Side Browser Isolation Models

The problem with the server-based model is that at some point you need to deploy some servers, the creators of client-side browser isolation technologies did the math and realized that there was no way that the server-side and virtualization based browser isolation platforms on the market could ever meet the core requirement.

With a client-side browser isolation platform all of the actual isolation is done on the users local machine, rather than on a remote server somewhere. Client-side solutions leverage the users local computers resources and remove the need for servers.

But the fundamental problem with client-side solutions is that they break the s***ecurity through physical isolation*** model by keeping all of the cyber risks on the same machine and isolating them locally. Client-side vendors ask us to trust that the attackers will not break out of the virtual machines they have created on the user's computer in order to isolate their browsers and the browsing activity away from the rest of the machine.

The problem with this is that attackers do actually break out of virtual machines all of the time in the real world which is the whole point of physically isolating the cyber risk in the first place, so that they are air-gapped away from your IT infrastructure.

While client-side solutions do indeed save you money over the long term by eliminating the need to deploy servers, you breach the security through physical isolation model.

Unfortunately for client-side vendors, their server-side competitors are significantly reducing their overhead by moving away from virtualization and towards containerization and when you need ten times less servers than you did before, suddenly the server based model starts to make a lot more financial sense.

In a market as large as the browser isolation market client-side solutions will undoubtedly find their fans, but judging from the market the current trend is pushing cyber risks away from your users computers and outside of your internal networks, this model is considered a best practice approach for browser isolation.

The Future Of Endpoint Security

Browser isolation is undoubtedly the future of endpoint security because it does not matter how fast the cyber threat landscape evolves, if your browsers and browsing activity is physically isolated away from your local machine and networks, 99% of the time it will stop the vast majority of malware and cyber threats dead in their tracks.

I say 99% of the time because it is important to acknowledge that out-of-band attacks that can breach the air-gap do exist, but they typically require close proximity to their target, or an insider within the organization in order for the attacks to be effective.

These kinds of attacks are just not scalable when it comes to groups of geographically dispersed internet users and these attacks also tend to be used by sophisticated nation-state hackers, not the kind of threat most organizations have to worry about.

For the vast majority of businesses and organizations browser isolation provides effective relief to the plague of web-based cyber attacks and malware. Browser isolation also allows organizations to take a proactive approach in dealing with these cyber threats by getting in front and ahead of them, rather than reacting to them when the threats are detected, or after a malware infection or breach has occurred.

This is why I say that browser isolation will become the future of endpoint security, it puts a stop to the browser-based malware and cyber attacks against end users at a time when our current security spend and security tools have completely failed us.

Browser isolation technologies represent a refreshingly effective way to rid yourself of the most common cyber risks affecting your employees as they use the internet.

When it comes to proactively enhancing your overall cybersecurity posture and shutting down the most common infiltration points on your network, nothing can beat browser isolation and in the future you can expect the model to become a bedrock of endpoint security as more and more organizations decide to outsource their cyber risk.

Outsourcing Your Cyber Risks

Ultimately outsourcing your cyber risks is what browser isolation is really all about and browser isolation best practice dictates that you outsource that cyber risk away from your internal networks and onto a platform that is specifically built to handle that risk.

When you rent a remote browser from a vendor, you are not really renting a service from them even though you may get a hosted remote browser in return, what you are actually doing is paying the vendor to take a specific set of cyber risks off your plate so that you can focus on securing the rest of your network and IT infrastructure.

I always thought of remote browser vendors as cyber risk outsource specialists, because are unique in that unlike any other kind of cybersecurity vendor, they personally take your cyber risks onto their own infrastructure and assume responsibility for them, something which carries with it a tremendous risk overhead for the vendor.

For many organizations such risk outsource services are a breath of fresh air, perhaps because they lack the resources internally to mount an effective cyber defense, or perhaps because it frees up their cybersecurity team to focus on securing something more important to your business than your users browsing activity.

It is my belief that when the cost of outsourcing cyber risk becomes inexpensive, this will trigger mass market adoption of browser isolation as an endpoint security solution over the long term. Right now the only thing slowing the market is a lack of customer awareness of the model and the expensive barrier to entry that is the price point.

What Does WEBGAP Mean?

According to the Urban Dictionary, a WEBGAP is a physical space between a user and their internet browser, one that isolates the user malware, viruses and ransomware.

It is also the name of the only vendor in the market that currently has a fully containerized and distributed browser isolation platform. The team at WEBGAP built the very first remote browser isolation platforms for the US federal government.

About The Author

Guise is the CEO and co-founder of WEBGAP and the co-developer of the Safeweb browser isolation model that he developed in collaboration with Robin Goldstone of Lawrence Livermore National Laboratory (LLNL.gov). Guise is also the founder of Secjuice, the only non-profit, independent & volunteer-run publication in infosec.

You can follow him on Twitter here: https://twitter.com/guisebule

You can connect with him on LinkedIn here: https://www.linkedin.com/in/guisebule/

You can also reach him by email using bule@webgap.io if you have any questions about anything in this white paper or about anything related to browser isolation.