MENLO SECURITY

Adaptive Clientless Rendering™

The Innovation Powering the Menlo Secure Enterprise Browser Solution

WHITE PAPER

Web browsers are among the most important applications in our business lives, yet they're also the most vulnerable to attack.

Most of us start our day in a web browser – as most organizations use a SaaS-based single-sign-on tool which displays the apps we use, most of which are web applications themselves. Many users have many if not dozens of browser tabs and groups by the end of the day. Gartner said[1] in 2023 that the web browser would become a super-app by 2027 but many would agree that it already is a super application, given that communications applications like Slack and Zoom and productivity apps like Microsoft 365 operate happily in our browsers.

The ubiquity of the browser throughout the workday has created a brand new set of risks for organizations of any size. Simple acts like clicking a link in an email that opens a browser page, or loading a web page in which malware could be embedded, may result in a compromised user endpoint, leading to malware installation, data theft, and penetration of corporate networks. The same browser is used for both work and casual web browsing. The boundaries between web applications are intentionally not strict, which is a double-edged sword: collaboration is ever easier, but so is oversharing and being led astray. Downloading malicious content via the web is easier for threat actors than even many security-trained users think. The modern workspace enables many attack vehicles for threat actors to target a user, including email, chat applications, text messages with QR codes, and poisoned search results. Unfortunately, even users who undergo yearly security training click on links that lead to infected sites (which can bypass Secure Web Gateways leveraging a LURE attack) or give away their credentials to bogus web forms that look legitimate.

At the same time, features keep being added to browsers to enable these ever more complex web applications. Each new feature represents a fresh attack surface as shown by a steady stream of browser vulnerabilities over the years. For example, recently browsers have added a number of AI integrations, including better support for GPU access via WebGPU, and with that, many opportunities for attackers.

So, today, how do threat actors compromise a browser to gain remote code execution? Sometimes, the victim is an unwitting participant: they get tricked into downloading and then executing a malicious payload with no vulnerabilities needed. In this scenario, it is crucial to have full visibility into the browser actions so that the deception can be stopped, both through AI-powered detection of malicious websites tricking users and through careful examination of downloaded content.

[1] G00782898, Emerging Tech: Security — The Future of Enterprise Browsers, April 2023

Other times, threat actors will target the above mentioned vulnerabilities with complex exploit chains. Gone are the days where a single vulnerability could lead to remote code execution. Such a scenario is vanishingly rare these days because modern endpoint operating systems and the browsers that run on them have added defenses to make exploiting single vulnerabilities more challenging. In particular, two widely used defenses: Data Execution Prevention (DEP/NX) and Address Space Layout Randomization (ASLR) can prevent simply injecting code via a buffer overflow and reusing existing code via Return-Oriented Programming (ROP), respectively. Instead, attackers must chain exploits targeting multiple vulnerabilities to gain unfettered code execution. Typically, three vulnerabilities will be needed: one to leak memory addresses (defeats ASLR), one to escape the JavaScript sandbox and gain native code execution, and one to escape the OS-level sandbox that constrains the process executing page content in multi-process browser architectures. A key insight is that each of these steps and the coordination between the successive steps requires the use of active content on the endpoint, in the form of attacker controlled JavaScript.

Active content enables an attacker to discern memory locations via address space disclosure, influence data layout via heap spray, and dictate code generation via JIT spray. These techniques are at the heart of most exploits today.

Menlo Secure Cloud Browser

The Menlo Secure Cloud Browser offers a solution to the security challenge posed by executing active content on the endpoint. It centers around the notion of a cloud-based, hardened digital twin of the desktop browser, that loads and runs pages, including all active content, inside a contained environment in the cloud with the goal of removing any potential browser infection from the endpoint. In most incarnations, the Secure Cloud Browser and the endpoint are separated by a secure channel using a minimal, highly optimized protocol designed to carry rendering updates to the endpoint and user input to the cloud, and nothing else. This browsing mode delivers defense against today's sophisticated zero-day exploits. Specifically, by running untrusted active content in the cloud and preventing it from probing and exfiltrating data from the endpoint, the Secure Cloud Browser prevents hacker exploitation of vulnerabilities that can readily bypass standard endpoint defenses.

Practical, Scalable Browser Security

Despite its compelling security benefits, cloud-based browser security must meet IT and end-user needs if it is to become a widely adopted security technology. To that end, we have identified five key requirements for a practical browser security solution:

1. **Clientless deployment** reduces IT burden by avoiding the need for endpoint software installs, and with it, the risk of destabilizing the endpoint. The clientless nature also enables easy enterprise-wide deployment via in-network proxy configuration, as well as fully centralized management of browsing policies and security updates across all devices—including personal devices—within the enterprise network.

2. **A native user experience,** in which users do not perceive any difference in web browsing, is critical for ensuring end-user productivity and buy-in. In particular, users should not have to alter the way they browse the web or be distracted by changes to their browser's behavior. Moreover, rendering speed and quality should be identical to native for a broad range of media types (text and video), and day-to-day operations such as printing and copy-paste should work in accordance with organizational zero trust policies. Geography shouldn't factor into the user experience either. Users need to be able to browse the web and access Software-as-a-Service (SaaS) platforms no matter where business takes them—whether across oceans or down the street in the local coffee shop.
3. **Scalability** is crucial. As IT organizations undergo cloud transformation, the number of web requests will only increase. Scaling to meet this demand needs to be effortless without the need for lengthy capacity planning.
4. Cloud-based browser security must also be **interoperable** with the rest of the security stack to ensure that all security policies are being met. This includes extending the broader architectural role of the Menlo Secure Cloud Browser to additional services, including SaaS governance and data loss prevention (DLP).
5. And finally, cloud-based browser security should **reduce operational costs** through better malware containment, fewer false positives, lower help-desk costs, and a decrease in the need to reimage damaged machines—because every expenditure in today's hyper-competitive business environment needs to pay for itself.

Meeting these requirements means solving the challenging problem of transparently delivering output from the Menlo Secure Cloud Browser to the existing endpoint browser without requiring endpoint modifications such as agents or special plug-ins.

Unfortunately, the traditional and most prevalent remoting technique—pixel mirroring, commonly used with virtual desktop intructure (VDI)–based video streaming—falls short, mainly because it treats the isolated web page as a bag of pixels to be mirrored with a client that has little understanding of what those pixels represent. The result is a one-size-fits-all approach that not only precludes adapting the remoting technique to the kind of content being displayed (e.g. text vs. video), but also slows down page load time and responsiveness by eliminating opportunities to harness the browser's hardware-accelerated rendering features, and compromises everyday workflow operations such as printing and copy-paste.

Recognizing the challenges posed by pixel-mirroring technology, several browser isolation solutions have traded off the clientless form factor for replacement browsers. Such solutions are seen as increasing IT security burdens caused by trouble tickets, endpoint disruption, and user friction, proving to be a significant barrier to broader deployment.

Adaptive Clientless Rendering

Patented Menlo Security Adaptive Clientless Rendering™ (ACR) is the core technology behind the Menlo Secure Enterprise Browser solution. In a clear departure from pixel-streaming, ACR combines a web-based delivery vehicle with a greater understanding of the isolated page to simultaneously enable clientless deployment and a native user experience.

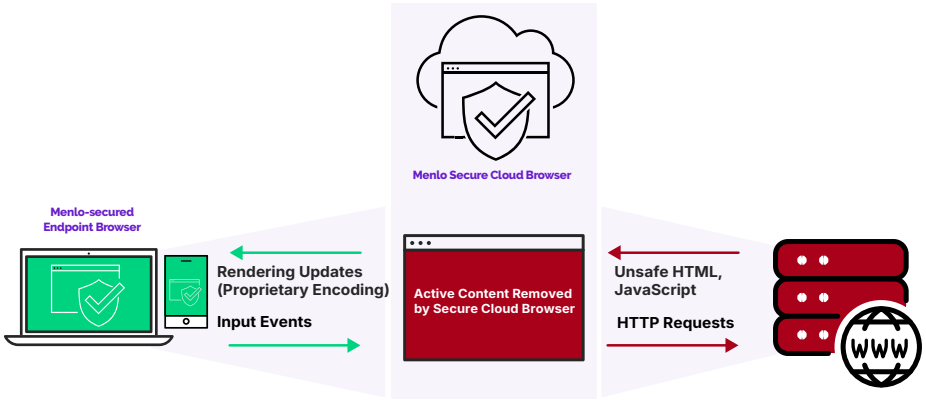

Figure 1: Adaptive Clientless Rendering overview. The existing endpoint browser loads a safe, transcoded version of the original page that interprets rendering updates coming from the isolated browser and relays input events back to it—all over a secure

Depicted in Figure 1, Adaptive Clientless Rendering involves two major components: the secure workspace environment and the Menlo Secure Cloud Browser.

The Menlo Secure Cloud Browser instance created for each endpoint browser delivers only safe, transcoded content to the endpoint browser. The desktop browser communicates with its hardened digital twin instance of the Menlo Secure Cloud Browser over a TLS-encrypted channel. The desktop component of ACR applies rendering updates from the Secure Cloud Browser and relays user inputs back to the cloud. Menlo ACR capitalizes on rapidly converging web standards and advances in browser engines to work accurately and efficiently regardless of which major browser or device is used on the endpoint.

The Menlo Secure Cloud Browser loads web pages on the endpoint's behalf. It sends rendering updates to the endpoint browser secured by Menlo in response to dynamic page changes and injects user inputs coming from the endpoint browser. Based on an always up-to-date version of the Chromium browser engine, the Secure Cloud browser inherits its security, stability, and feature set. However, since no browser engine is immune to infection, the Menlo Secure Enterprise Browser solution operates under the assumption that active hardened digital twin instances of the Secure Cloud Browser could eventually be infected. Thus, as a key step in protecting the Secure Cloud Browser as well as end users, the Menlo architecture employs frequent disposal of Secure Cloud Browser instances along with multi-level container isolation to avoid the spread of malware that is increasingly being referred to as HEAT, or highly evasive, adaptive threat attacks. Such attacks attempt to morph internally to avoid detection and remain persistent, and, seek paths to move laterally between applications and operating systems. See this page for more information on HEAT attacks.

Multiple Modes to Deliver a Seamless User Experience

Understanding the power of Adaptive Clientless Rendering requires an appreciation of web page rendering. The Document Object Model (DOM) structure is a dynamic, browser-internal representation of how a web page should be rendered. The browser combines DOM with Cascading Style Sheets (CSS) to render page content. The key to the ACR transparent user experience is the ability of ACR to create and maintain a DOM structure equivalent to what the active content that has been removed by Secure Cloud Browser. Reconstructing a DOM enables most of the rendering work, including interactive animations such as scrolling, to be performed solely on the endpoint browser using its built-in GPU-optimized machinery. Moreover, a reconstructed DOM exposes semantics of the content being rendered so that the full browser feature set—including copy-paste, find in page, printing, and password managers—continues to function.

ACR employs two distinct reconstruction modes that are dynamically selected at page granularity based on a variety of factors, including the endpoint device used and the content of the page.

Mode 1: DOM Mirroring

Intuitively, the goal of DOM Mirroring is to mirror only the benign DOM information to the endpoint browser. To reflect DOM changes on the client, the Secure Cloud Browser actively monitors the currently loaded page's DOM tree. These updates are applied to the local DOM using the standard DOM API available in all browsers.

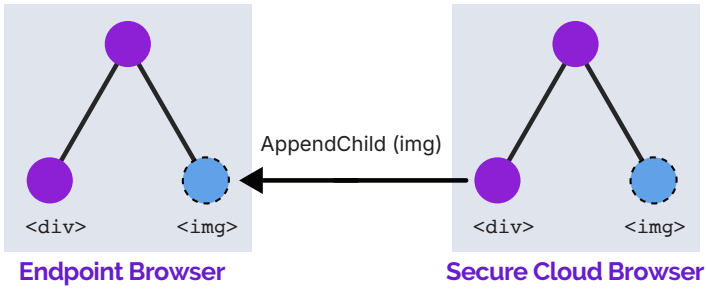

Figure 2: DOM Mirroring allows the mirroring of the DOM tree and the resources to the endpoint browser.

Adaptive Transcoding: DOM Mirroring confers distinct advantages over pixel streaming approaches, primarily owing to the selective exposure of DOM elements to the client. A key benefit is that it enables the selection of remoting strategy at DOM element granularity, so nonactive safe elements are left as is and active unsafe elements are either dropped altogether or are replaced with a safe, transcoded variant that is best suited for the element's media type.

Rendering and Workflow Offloading: Central to providing a truly transparent user experience, DOM Mirroring leverages the capabilities of the endpoint browser—resulting in visible benefits such as fast page loads, smooth scrolling and animations, and crisp, high-quality HTML5 video playback. The semantically aware rendering of DOM Mirroring also enables the client browser to render a truly native look and feel, regardless of endpoint browser or platform.

Finally, DOM Mirroring avoids disruption to workflow operations such as copy-paste, find-replace, and printing. Copy-paste, for example, is difficult to emulate in a truly native fashion using pixel streaming because of browser-enforced security restrictions on asynchronous clipboard manipulation. Printing, too, is encumbered by the endpoint browser's view of the page as a block of pixels, as opposed to a document that can be reflowed to accommodate any output device. In contrast, DOM Mirroring provides the endpoint browser's existing workflow mechanisms with all the information it needs to do its job, so emulation is not needed.

Mode 2: Smart DOM

As business moves to the cloud, and as user access to SaaS platforms and web apps through mobile platforms becomes mission critical, the rendering engine in the Menlo Secure Cloud Browser adapts to these trends. The proliferation of mobile devices introduces the need to be both power and network efficient when remoting content to the endpoint. Menlo Security Smart DOM takes these new technical requirements into account.

Rather than mirror the DOM tree present in the Secure Cloud Browser, Smart DOM generates a different yet equivalent DOM tree on the endpoint browser using low-level rendering data structures provided by the Menlo Secure Cloud Browser *compositor* subsystem. The unique approach to DOM reconstruction used by Smart DOM confers several benefits:

- **Smart DOM ACR renders pages accurately across a wide range of browsers,** thus offering a more consistent user experience across browsers old and new, as well as across desktop and mobile devices. Smart DOM avoids cross-browser incompatibilities using patented techniques
- Smart DOM is **bandwidth efficient**. Unlike pixel streaming, Smart DOM avoids duplicate network transfers
- As with DOM, **Smart DOM preserves the native user experience.** Smart DOM renders quickly on the endpoint, naturally harnessing the browser's GPU acceleration to enable 60FPS scrolling, pinch-zoom, and animations across desktop and mobile devices alike. Moreover, Smart DOM facilitates the accurate emulation of native-supporting functionality such as copy-paste, find-replace, printing, mobile text input, endpoint local fonts, and native page widgets
- **Smart DOM renders on the endpoint.** Smart DOM mirrors no active content, thus exposing minimal attack surface.

This new way of safely mirroring content on users' browsers ensures that Menlo ACR technology is adaptable in the truest sense. New modes can be added as browsers and web development tools evolve. **Menlo intelligently optimizes which mode to use and sets Smart DOM mode as the default for mobile.** Together, the two rendering modes are delivered by a cloud service that continuously advances without impacting service uptime.

This allows the Menlo Secure Cloud Browser to seamlessly and safely render web content to users' devices while increasing the accuracy of the native browser experience, scaling to a wider array of different media, and decreasing the amount of bandwidth that cloud-based browser security requires. Most importantly, the user's browsing experience is not changed at all. Users can continue to browse the web, access web apps, and work online as they have always done, with no impact on performance.

Security

The guiding principle behind the ACR security model is the recognition that active content execution is the attacker's key to successful endpoint defense evasion. Preventing active content execution on the endpoint results in little hope that an exploit will bypass existing defenses, regardless of what a potentially compromised Secure Cloud Browser sends to the client. This principle drives ACR use of multiple security mechanisms that deliver strong defenses against even the most determined adversaries.

The first mechanism, **active content blocking and transcoding**, ensures that active content never intentionally gets sent to the endpoint. In DOM Mirroring mode, the Secure Cloud Browser filters all incoming DOM elements, attributes, and CSS against an allow list. For instance, <script> elements and onclick attributes are dropped, while <object> elements are replaced with a safe remoting widget that displays the transcoded, real-time output of the plug-in window as it is rendered by the Secure Cloud Browser.

Smart DOM requires no special filtering mechanism as it sends only compositor-level structures that are guaranteed by design not to contain DOM or CSS state. As an added layer of protection, the Menlo architecture employs a **Content Security Policy** at its strictest setting (no inline script, no plug-in) to ensure that the endpoint browser blocks all active content executions.

To appreciate the second security mechanism, it's important to consider that each instance of the Menlo Secure Cloud Browser is in fact a browser, albeit one that is container-isolated, relatively short-lived, and rigorously maintained (Chromium updates). But any browser can be compromised by a sufficiently determined threat actor. Therefore, each instance of the Menlo Secure Cloud Browser is surrounded by services that together fully contain browser compromises.

First, Menlo software operating on the secured endpoint browser is delivered not from the Secure Cloud Browser but, rather, a static resource server operating elsewhere in the Menlo Cloud, ensuring client code authenticity. Then **protocol checking and enforcement** ensures that the Menlo-secured endpoint browser will refuse to download or execute active content from any instance of the Secure Cloud Browser. In particular, all rendering updates coming from the Secure Cloud Browser are required to be in a canonical format. For example, in DOM Mirroring mode, DIV element updates must have the "div" tag; the endpoint does not accept any other string variant, even those including strange character codes that may be interpreted unexpectedly by the endpoint browser. Second, the endpoint verifies that outgoing messages adhere to a simple user-input protocol; e.g., "click button 1," "keypress code 45," or "scroll to 45." The aggregated security architecture ensures that even a potentially compromised Secure Cloud Browser instance has no channel with which to probe the endpoint for vulnerabilities or to exfiltrate information that could be used to bypass standard endpoint defenses.

Conclusion

The browser feature set continues to expand with active content firmly remaining the predominant vector for exploitation. The Menlo Secure Enterprise Browser solution shields users from these future threats by running active content away from the endpoint. However, to reach its true potential, cloud-based browser security faces the challenge of providing both clientless deployment and a fully transparent user experience. Adaptive Clientless Rendering™—the proven remoting technology at the core of the Menlo Secure Enterprise Browser solution—meets this challenge by adaptively and safely reconstructing a full-fledged DOM on the endpoint browser, thus enabling native rendering and the full scope of native browser functionality.

With mechanisms to ensure that active content never executes on the endpoint browser, ACR defends against zero-day threats while providing a clientless and native browsing experience.

To learn more about securing the ways people work, visit menlosecurity.com or email us at ask@menlosecurity.com.

About Menlo Security

Menlo Security eliminates evasive threats and protects productivity with the Menlo Secure Cloud Browser. Menlo delivers on the promise of cloud-based security—enabling zero trust access that is simple to deploy. The Menlo Secure Cloud Browser prevents attacks and makes cyber defenses invisible to end users while they work online, reducing the operational burden on security teams.

Menlo protects your users and secures access to applications, providing a complete enterprise browser solution. With Menlo, you can deploy browser security policies in a single click, secure SaaS and private application access, and protect enterprise data down to the last mile. Secure your digital transformation with trusted and proven cyber defenses, on any browser.