ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 14, Issue 3, pp: (77-80), Month: July – September 2026, Available at: www.researchpublish.com

The Curious Case of the Slow Logon with a Black Screen

Junaid Ahmed Jan

DOI: https://doi.org/10.5281/zenodo.22646673

Published Date: 07-September-2026

Abstract: **A recurring 3–4-minute black-screen delay during Windows 11 logon was traced to a security overlay application that injected itself early in the logon sequence. Procmon and network traces revealed that the application repeatedly attempted to contact a backend server for configuration data, blocking Explorer.exe from loading. When the backend was unreachable, long TCP timeout cycles caused the entire logon to stall. By reproducing the issue through hosts-file redirection, the investigation confirmed the dependency as the root cause. The case underscores the need for fast-fail logic, offline capability, and non-blocking initialization for any component operating in the Windows logon path.**

Keywords: **black-screen, network traces, hosts-file redirection, offline capability.**

I. INTRODUCTION

Windows logon delays are notoriously difficult to troubleshoot. They often hide behind layers of authentication, profile loading, GPO processing, and third-party agents that hook into the logon sequence. But every once in a while, an investigation uncovers a root cause so unexpected and so deceptively simple that it becomes a case study worth documenting.

This is one such case.

The Symptom: A 3–4 Minute Black Screen After Entering PIN on Windows 11

Consistent pattern:

• After entering the Windows Hello PIN.

• The screen went black.

• Nothing appeared for 3–4 minutes.

• Eventually, the desktop loaded normally.

No crashes. No errors. No event logs pointing to authentication issues. Just a long, silent pause.

This behavior was reproducible across multiple systems.

2. The Initial Clues: Applications in the Startup Path

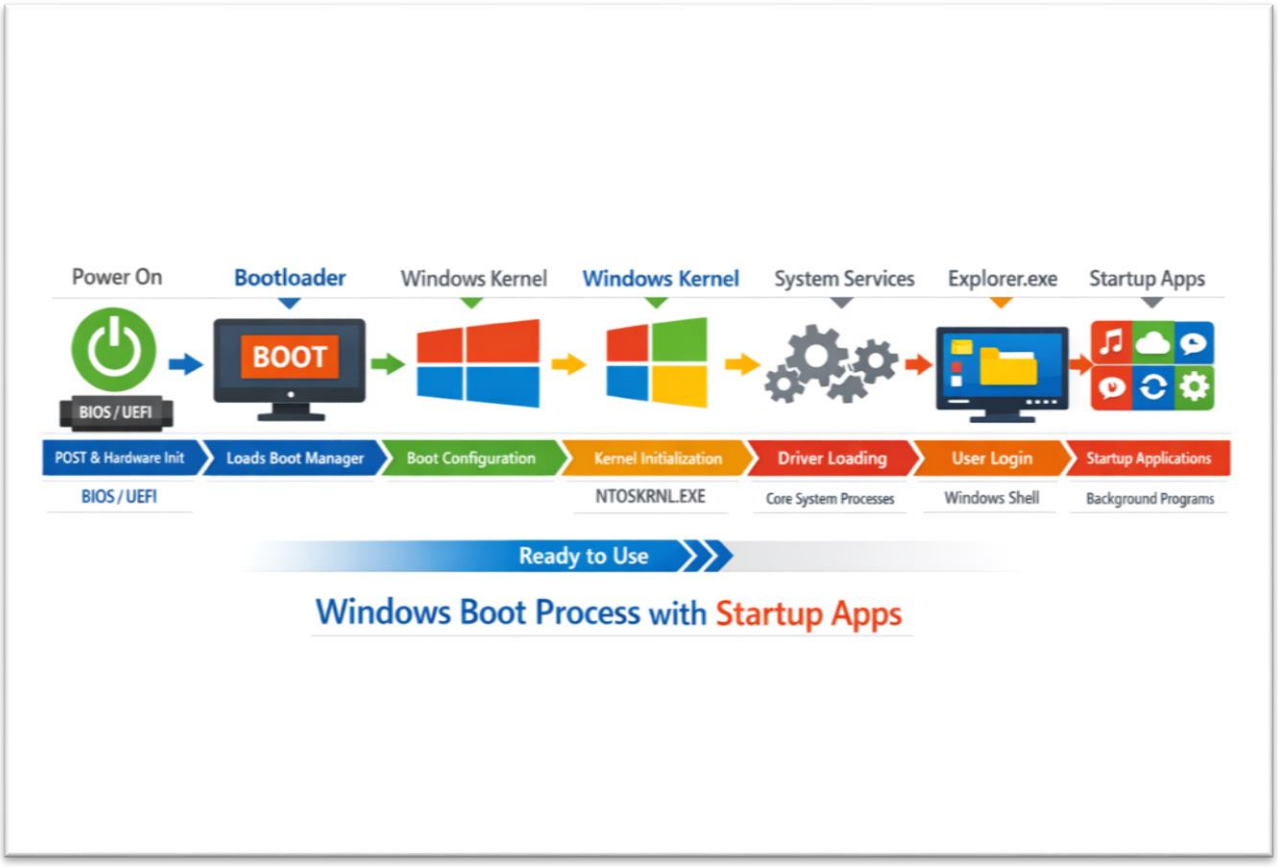
The affected systems had a security-oriented screen overlay application installed. Its purpose was to overlay encoded session information such as username, timestamp, or device ID on the desktop to deter data leakage.

This application injected itself early in the logon process, right after credential validation but before the shell (Explorer.exe) fully initialized.

Under normal conditions, the application worked properly.

But during the problematic logons, there was a 3–4-minutes of delay, strongly suggesting it was part of the bottleneck.

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 14, Issue 3, pp: (77-80), Month: July – September 2026, Available at: www.researchpublish.com



Deep Dive: Collecting Procmon and Network Traces

To understand what was happening behind the black screen, two key data sources can be collected:

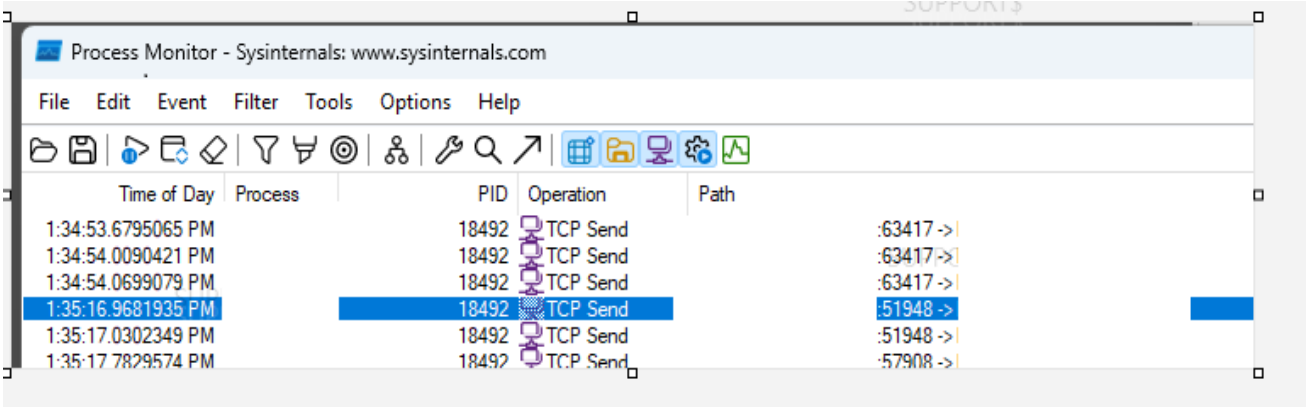
1. Procmon Traces

• The application executable launched immediately after credential acceptance.

• It attempted to read configuration files.

• It then initiated outbound network calls before rendering the overlay.

• The process repeatedly retried network connections for several minutes.



2. Network Traces

These showed -

• DNS queries for the watermark backend.

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 14, Issue 3, pp: (77-80), Month: July – September 2026, Available at: www.researchpublish.com

• TCP SYN & SYN RE-TRANSMIT attempts to IP addresses.

• Long timeout intervals before fallback logic triggered.

Protocol Name	Description	Conv Id
TCP	TCP:Flags=......S., SrcPort=55011, DstPort=HTTPS(443), PayloadLen=0, Seq=2532849970, Ack=0, Win=65535 (Nego...	{TCP:126, IPv4:125}
TCP	TCP:Flags=......S., SrcPort=51661, DstPort=HTTPS(443), PayloadLen=0, Seq=2630681864, Ack=0, Win=65535 (Nego...	{TCP:127, IPv4:125}
TCP	TCP:Flags=......S., SrcPort=50898, DstPort=HTTPS(443), PayloadLen=0, Seq=3151509679, Ack=0, Win=65535 (Nego...	{TCP:128, IPv4:125}
TCP	TCP:[SynReTransmit #570]Flags=......S., SrcPort=55011, DstPort=HTTPS(443), PayloadLen=0, Seq=2532849970, Ac...	{TCP:126, IPv4:125}
TCP	TCP:[SynReTransmit #571]Flags=......S., SrcPort=51661, DstPort=HTTPS(443), PayloadLen=0, Seq=2630681864, Ac...	{TCP:127, IPv4:125}
TCP	TCP:[SynReTransmit #577]Flags=......S., SrcPort=50898, DstPort=HTTPS(443), PayloadLen=0, Seq=3151509679, Ac...	{TCP:128, IPv4:125}
TCP	TCP:[SynReTransmit #570]Flags=......S., SrcPort=55011, DstPort=HTTPS(443), PayloadLen=0, Seq=2532849970, Ac...	{TCP:126, IPv4:125}
TCP	TCP:[SynReTransmit #571]Flags=......S., SrcPort=51661, DstPort=HTTPS(443), PayloadLen=0, Seq=2630681864, Ac...	{TCP:127, IPv4:125}
TCP	TCP:[SynReTransmit #577]Flags=......S., SrcPort=50898, DstPort=HTTPS(443), PayloadLen=0, Seq=3151509679, Ac...	{TCP:128, IPv4:125}
TCP	TCP:[SynReTransmit #570]Flags=......S., SrcPort=55011, DstPort=HTTPS(443), PayloadLen=0, Seq=2532849970, Ac...	{TCP:126, IPv4:125}
TCP	TCP:[SynReTransmit #571]Flags=......S., SrcPort=51661, DstPort=HTTPS(443), PayloadLen=0, Seq=2630681864, Ac...	{TCP:127, IPv4:125}
TCP	TCP:[SynReTransmit #577]Flags=......S., SrcPort=50898, DstPort=HTTPS(443), PayloadLen=0, Seq=3151509679, Ac...	{TCP:128, IPv4:125}
TCP	TCP:Flags=......S., SrcPort=58132, DstPort=HTTPS(443), PayloadLen=0, Seq=459855041, Ack=0, Win=65535 (Negoti...	{TCP:142, IPv4:125}
TCP	TCP:Flags=......S., SrcPort=61943, DstPort=HTTPS(443), PayloadLen=0, Seq=4207383469, Ack=0, Win=65535 (Nego...	{TCP:143, IPv4:125}

The pattern was unmistakable:

The application was blocking the logon sequence while waiting for backend configuration.

Root Cause: Backend Dependency Without Timeout Safeguards

The application was designed to fetch:

• Overlay configuration

• Encoding rules

• Session identifiers

• Policy updates

It contacted a backend server during every user logon.

However, if the backend was:

• unreachable,

• misconfigured,

• or the network path was broken

The application did not fail fast.

Instead, it waited for multiple connection retries, each with long TCP timeout intervals. During this time, the user saw only a black screen.

This explained the intermittent nature of the issue:

- When the backend was healthy, logon was instant.
- When the backend was down or unreachable, the delay appeared.

Reproducing the Issue: Host File Manipulation

To validate the hypothesis, the test system's hosts file was modified:

• The backend hostname was mapped to a fake, non-responsive IP address.

• This forced the watermark application to attempt connections that would never succeed.

The result was a perfect reproduction of the issue:

• Black screen

• 3–4 minute delay

• Desktop finally loading once the application gave up

This confirmed that the logon delay was directly tied to the application's network dependency.

Why This Caused a Black Screen Instead of a Visible Error

Several architectural factors contributed:

• The application injected itself before Explorer.exe fully initialized.

• It ran in a privileged session with UI hooks.

• The system waited for the overlay to initialize before continuing the logon sequence.

• The application lacked asynchronous or timeout-aware design.

In short:

The OS was waiting for the application to finish its startup routine, and the application was waiting for a backend response.

2. RECOMMENDATION

Fast-Fail Logic and Offline Mode

• Shorter connection timeouts

• Graceful fallback to cached configuration

• Offline mode when backend servers are unreachable

• Non-blocking initialization, allowing the desktop to load even if the overlay is delayed

Lessons Learned

This case highlights several important principles for enterprise IT and security engineering:

1. Any application in the logon path / Startup up entry must fail fast.

Long network timeouts can cripple the user experience.

2. Security overlays and agents must support offline operation.

Enterprise networks are not always reliable.

The hosts-file manipulation provided undeniable proof of the root cause.

3. CONCLUSION

What began as a mysterious black-screen delay turned out to be a classic example of a well-intentioned security tool inadvertently degrading system usability due to poor timeout handling.

This case serves as a reminder:

In the Windows logon sequence, even small components can have massive impact if they block the critical path.