SecCore

# ITSecX 2025

## Unconventional Initial Access Vectors

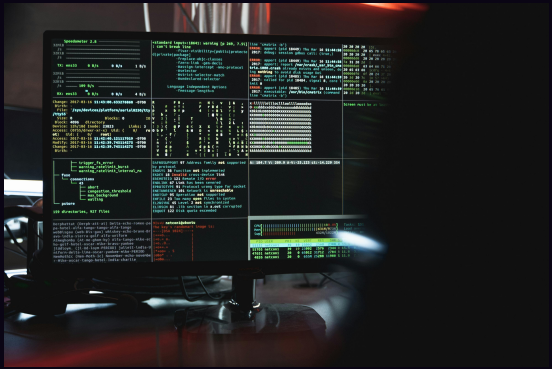
# About Us

**Patrick Pongratz**
SecCore GmbH
CEO & Co-Founder
Penetration Tester
Red & Purple Teamer
patrick.pongratz@seccore.at

**Benjamin Floriani**
SecCore GmbH
CEO & Co-Founder
Penetration Tester
Red & Purple Teamer
benjamin.floriani@seccore.at

# Initial Access Vectors

- Various methods to gain initial access to a target environment:
  - Physical
  - Technical
  - Social
- Usually not necessary in a classic pentest engagement
- Very important in Red Team engagements (and real attacks)
- Focus on uncommon methods



# Physical

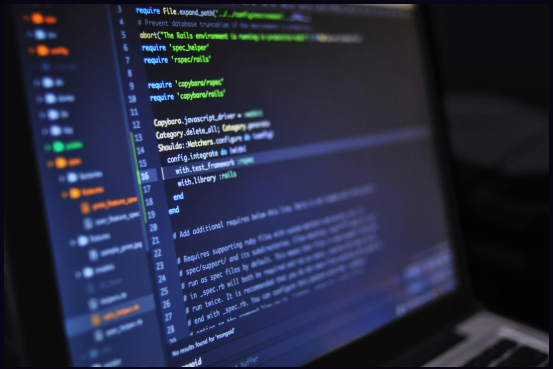
- USB Devices
  - Infected files (executables, Office documents, etc.)
  - BadUSB (e.g. RubberDucky)
  - Keyloggers
- Hardware Implants
- Dumpster Diving
- Theft / Borrowing ;)
- etc.

Downside: Physical access required

# Technical

- Public Facing Services
  - Known Vulnerabilities (e.g. RCE)
  - Misconfigurations (e.g. open DBs or storage accounts)
  - Weak Credentials (e.g. brute-force / password spraying)
  - This also includes cloud services!
- Supply Chain Attacks

Downside: Time-intensive and noisy



# Social

- Usually the most effective method
- Often combined with physical or technical methods
- Classic Social Engineering
  - Phishing
  - Vishing (Voice Phishing)
  - Smishing (SMS Phishing)
  - Quishing (QR Code Phishing)
  - In-person (e.g. tailgating, IT support disguise)
- Can be done very targeted (spear-phishing)

Downside: Need to bypass awareness and defenses

# Awareness and Defenses

- Awareness trainings and technical defenses (e.g. email filtering) are improving
- Classic phishing techniques such as links and well known malicious attachments are less effective:
  - Users know to be cautious of external links
  - Possible malicious attachments (e.g. .exe, .hta, .docm) are often blocked
  - Users are trained to recognize common phishing techniques
  - MFA is widely adopted

# How can we stay ahead?

# An Unconventional Vector

Our Initial Access Vector needs to:

- Bypass technical defenses (e.g. email filtering, sandboxing)
- Bypass user awareness (e.g. avoid links and known malicious attachments)
- Not require physical access
- Bypass Multi-Factor Authentication (MFA)

Easy, right?

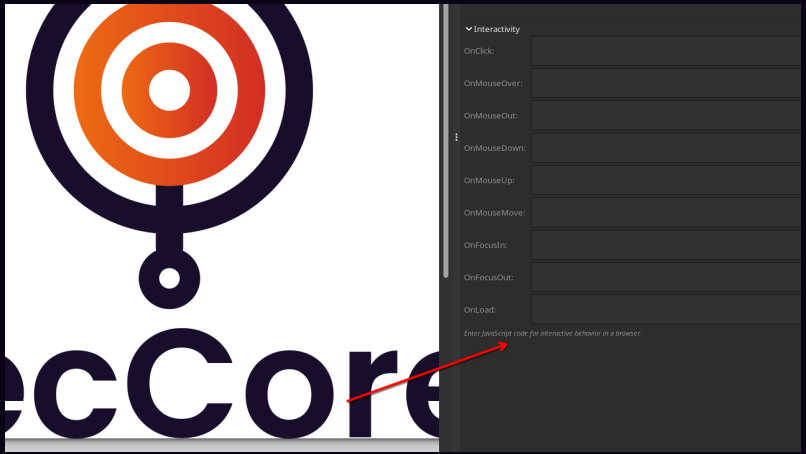
# An Unconventional Vector

When we founded our company, one of the first things we did was to try and find a good logo.

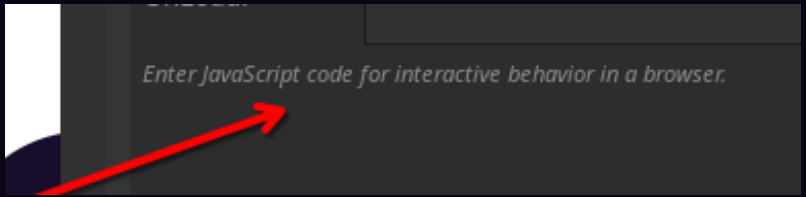
For logos, there is basically one file format that is used almost exclusively: **Scalable Vector Graphics (SVG)**.

Those vector images can be scaled to any size without losing quality.
SVG-files are basically XML-files that describe the image.
And there is one feature of SVG that makes it interesting for us:
**SVG supports embedding JavaScript code!**

# An Unconventional Vector



# An Unconventional Vector

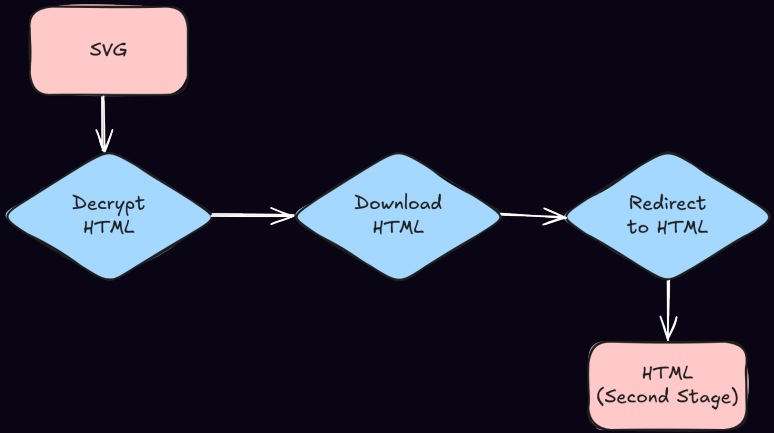


# An Unconventional Vector

However, there are some challenges:

- There is no Document Object Model (DOM) in SVG
- So creating a legitimate looking phishing page inside of an SVG is not possible

# Our Approach



# The Second Stage Payload

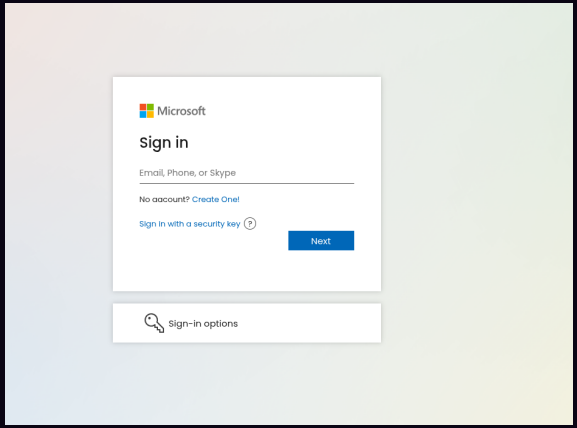
# The Second Stage

The second stage payload can be any HTML file (now with a working DOM). This means that we can create a very realistic phishing page that looks exactly like the real login page of our target.

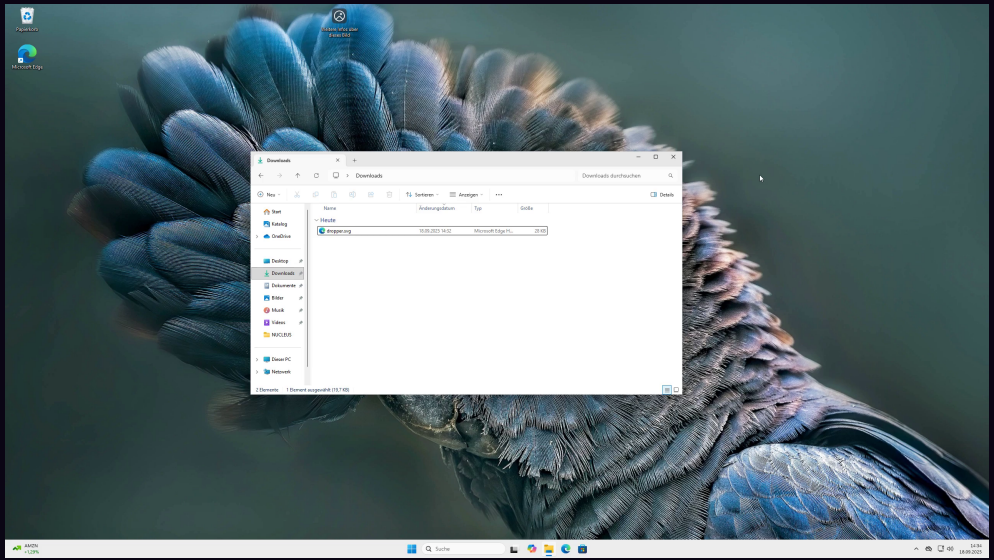
Since the HTML file is loaded locally, it will also bypass URL filtering and website sandboxing techniques.

When credentials are submitted, they will be sent straight to our backend server, where the MFA challenge will be triggered.

# The Second Stage



# Live Demo:
# Creating and opening the SVG file


Papierkorb
Microsoft Edge
Downloads
Downloads durchsuchen
Neu
Sortieren
Anzeigen
Details
Start
Katalog
OneDrive
Desktop
Downloads
Dokumente
Bilder
Musik
Videos
NUCLEUS
Dieser PC
Netzwerk
Name
Änderungsdatum
Typ
Größe
Heute
dropper.org
16.09.2025 14:52
Microsoft Edge H...
20 KB
2 Elemente
1 Element ausgewählt (19,7 KB)
Suche

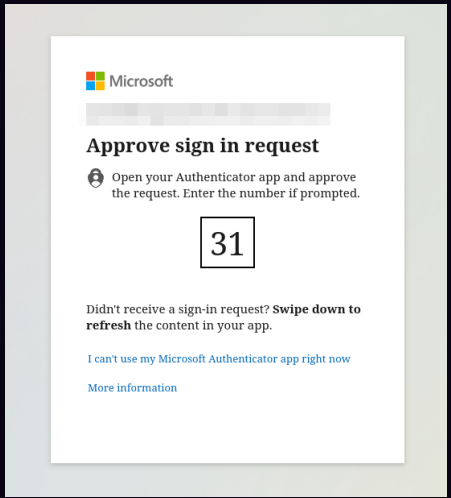
# The Backend

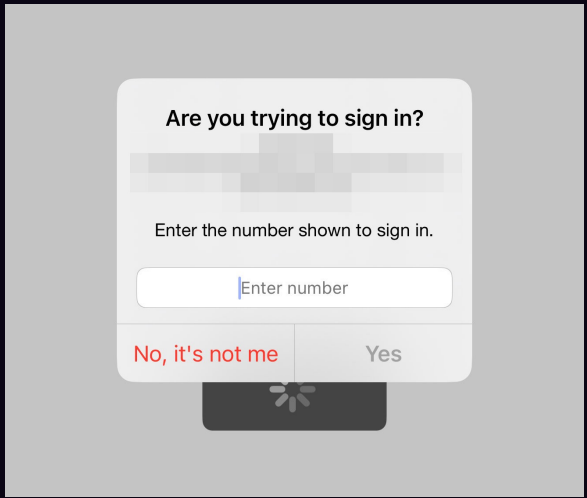
Now that we have everything we need to trick the user into submitting their credentials, we need to handle the backend.

One of the hardest challenges today (for attackers) is to bypass MFA.
We need to make sure that we can trigger the MFA challenge and receive the code. Then, we need to relay the MFA code to the user, so that they can enter it into their Authenticator app.

For this talk, we will only be showing the attack on Microsoft 365 / Entra ID using the default Microsoft Authenticator app. In our experience, this is the most common setup in corporate environments.

# The Backend


Microsoft
Approve sign in request
Open your Authenticator app and approve the request. Enter the number if prompted.
31
Didn't receive a sign-in request? Swipe down to refresh the content in your app.
I can't use my Microsoft Authenticator app right now
More information


Are you trying to sign in?
Enter the number shown to sign in.
Enter number
No, it's not me
Yes

# The Backend

We do not need or want phishing frameworks, since they are often easy to detect, and making them stealthy is more work than just doing it ourselves.

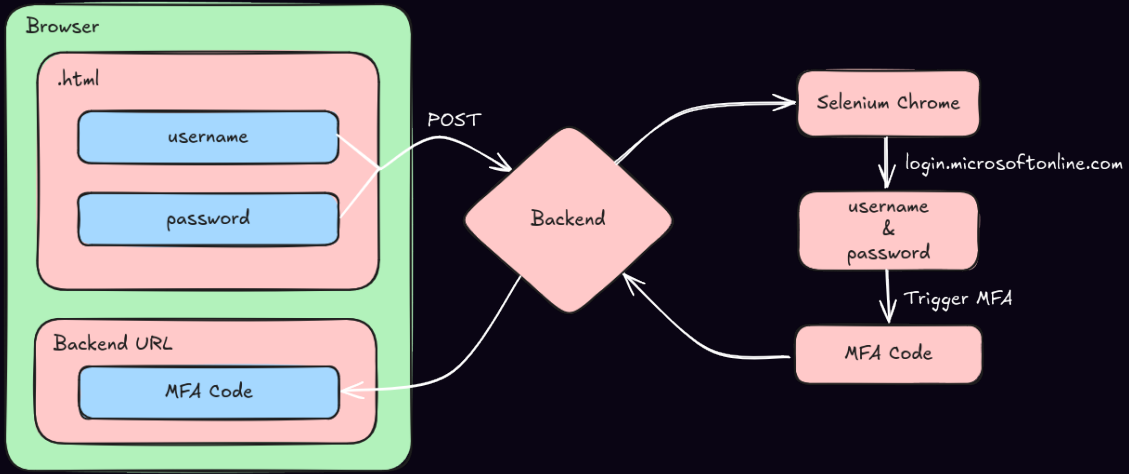
We also do not need a full web application stack, since we only need to receive credentials and send back the MFA code.

We also tailor it to every engagement, so we do not need a generic solution.

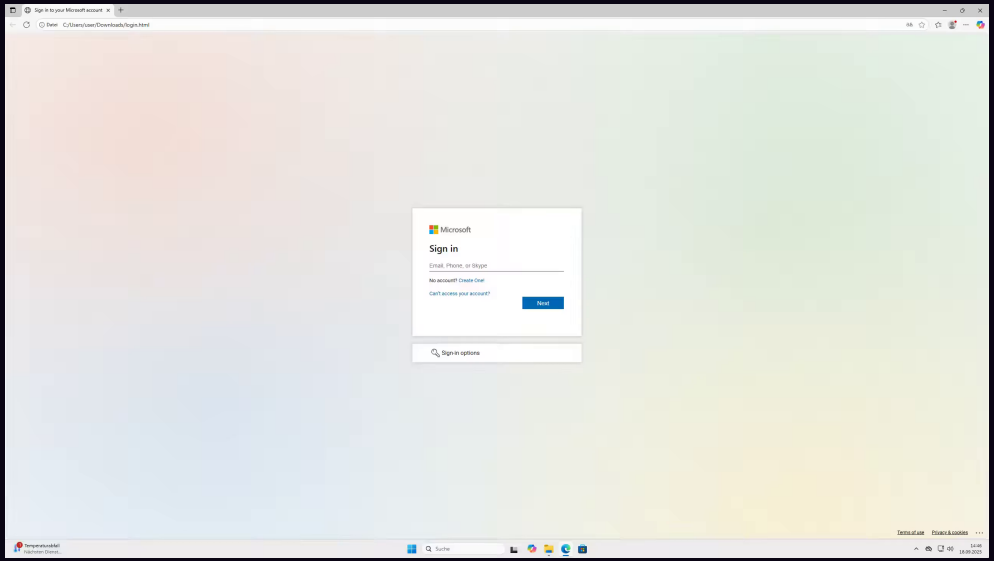
This makes our backend very portable:

- Can be hosted as Azure container instance, serverless function, etc.
- Only username & password are sent to the backend
- No other internet communication from the SVG or HTML file

# Overview of the Attack Flow



# Live Demo: Entering Credentials and Bypassing MFA


Sign in to your Microsoft account
Datei C:/Users/user/Downloads/login.html
Microsoft
Sign in
Email, Phone, or Skype
No account? Create One!
Can't access your account?
Next
Sign-in options
Terms of use
Privacy & cookies
Suche

# Wrapping Up

- ✔ SVG attachment serves as first stage payload
- ✔ Those files are not ”known malicious” yet
- ✔ SVG file drops an HTML file as second stage payload
- ✔ Second stage HTML file hosts a phishing page fully locally
- ✔ Only username & password are sent to the backend
  - No other internet communication from the SVG or HTML file
  - This makes the backend very portable
  - Successful login triggers an MFA challenge
  - MFA code is relayed to the user by the backend
  - Successful MFA verification gives us access to the account via Chrome browser session
- ✔ First stage, second stage and backend can have Anti-Sandboxing techniques (they do in real engagements)

# ! Other Possibilities

- Redirect to phishing page
- Drop a malicious file directly (e.g. .hta, .docm, .exe, etc.)
  - We cannot launch this file directly from the browser, but we can trick the user into opening it
  - Drop a fake installer.exe or invoice.doc
  - Use social engineering to convince the user to open it
  - Or just wait

# Mitigation

- Train users to be cautious of suddenly appearing login prompts - even if they are unfortunately very common nowadays
- Use technical defenses to block SVG files with embedded JavaScript
- Use Conditional Access policies to block logins from unfamiliar locations or devices
- Use risk-based MFA solutions that can detect unusual login behavior

# Questions?

# Thank You!

Interested in high quality security assessments? Reach out to us!

**SecCore**

https://seccore.at

contact@seccore.at

seccore-gmbh

SecCoreGmbH