**PENETRATION TEST REPORT**

Metal Finishing Calculator v3.0.0

Full Technical Security Assessment

|  |
| --- |
| **CONFIDENTIAL**  This document contains sensitive security information. Distribution is limited to authorized personnel under NDA. Unauthorized disclosure is prohibited. |

|  |  |
| --- | --- |
| **Field** | **Value** |
| Prepared for: | Psyrcuit LLC |
| Report Date: | January 30, 2026 |
| Report Reference: | MFC-SEC-2026-001-FULL |
| Classification: | Internal Use / NDA Required |
| Document Version: | 1.0 |

## **4.7 FINDING-007: Add-On System Cryptographic Verification Review**

|  |  |
| --- | --- |
| **Attribute** | Detail |
| **Finding ID** | FINDING-007 |
| **Severity** | INFORMATIONAL |
| **Status** | VERIFIED – Controls operating as designed |
| **Component** | AddonManager (main.js lines 1299–1770) |
| **Description** | Review of the add-on system’s cryptographic signature enforcement using Ed25519 elliptic curve signatures with SHA-256 content hashing. |

**Security Tests Performed:**

|  |  |
| --- | --- |
| **Test** | Result |
| **1. Install unsigned add-on** | BLOCKED – Flagged as unverified, controls disabled |
| **2. Install add-on with forged signature** | BLOCKED – Signature verification failed |
| **3. Modify signed add-on content post-install** | BLOCKED – Hash mismatch detected at next launch |
| **4. Rename signed add-on directory** | BLOCKED – Re-verification triggered, signature invalid for new path |
| **5. Directory traversal via add-on paths** | BLOCKED – Path sanitization prevents escape |
| **6. Network exfiltration from add-on code** | BLOCKED – CSP and disabled Node.js prevent network access |
| **7. Node.js API access from add-on renderer** | BLOCKED – nodeIntegration: false enforced |
| **8. Privilege escalation to main process** | BLOCKED – contextIsolation: true prevents bridge access |
| **9. Bypass signature check via binary modification** | OUT OF SCOPE – Requires application binary tampering (voiding warranty) |

Conclusion: The add-on system’s Ed25519 signature verification chain is correctly implemented with defense-in-depth enforcement at installation, launch, and renderer load. No bypass vectors were identified within the application’s trust boundary.

|  |  |
| --- | --- |
| **Recommendation** | No action required. Signature enforcement operating as designed. Future consideration: add certificate pinning for add-on update distribution channels. |

# **Table of Contents**

1. Executive Summary

2. Assessment Overview

3. Methodology

4. Findings Detail

5. Risk Assessment Matrix

6. Remediation Summary

7. Security Architecture Review

8. Recommendations

9. Appendix A: Tool Output

10. Appendix B: Code Changes

# **1. Executive Summary**

## **1.1 Overview**

A comprehensive security assessment was conducted on the Metal Finishing Calculator (MFC) v3.0.0 Electron application. The assessment utilized automated static analysis tools combined with manual code review to identify security vulnerabilities and deviations from Electron security best practices.

## **1.2 Key Findings**

|  |  |  |
| --- | --- | --- |
| **Severity** | **Count** | **Status** |
| Critical | 0 | N/A |
| High | 1 | Remediated |
| Medium | 5 | 4 Remediated, 1 Accepted |
| Low | 0 | N/A |
| Informational | 1 | Reviewed |

## **1.3 Overall Risk Rating**

|  |  |
| --- | --- |
| **Assessment Phase** | **Risk Level** |
| Before Remediation | Medium-High |
| After Remediation | Low |

## **1.4 Conclusion**

The Metal Finishing Calculator v3.0.0 has been hardened against common Electron security vulnerabilities. All critical and high-severity findings have been remediated. The application now implements defense-in-depth security controls appropriate for a standalone desktop calculator application handling sensitive business data.

# **2. Assessment Overview**

## **2.1 Scope**

|  |  |  |
| --- | --- | --- |
| **Component** | **Included** | **Notes** |
| Electron Main Process (main.js) | ✓ | Full analysis |
| Preload Script (preload.js) | ✓ | API exposure review |
| Renderer Process (index.html) | ✓ | CSP and DOM security |
| IPC Communication | ✓ | Channel security |
| Build Configuration (package.json) | ✓ | Dependency analysis |
| Third-Party Dependencies | ✓ | npm audit |

## **2.2 Test Environment**

|  |  |
| --- | --- |
| **Item** | **Details** |
| Application Version | Metal Finishing Calculator v3.0.0 |
| Electron Version | 40.1.0 (post-remediation) |
| electron-builder Version | 26.6.0 (post-remediation) |
| Operating System | Windows 10/11 x64 |
| Analysis Platform | Ubuntu 24.04 (GitHub Codespaces) |

## **2.3 Tools Used**

|  |  |  |
| --- | --- | --- |
| **Tool** | **Version** | **Purpose** |
| Electronegativity | 1.10.2 | Electron security static analysis |
| npm audit | 10.x | Dependency vulnerability scanning |
| Manual Code Review | N/A | Security pattern analysis |

# **3. Methodology**

## **3.1 Assessment Approach**

The security assessment followed a white-box methodology with full access to source code:

### **Phase 1: Reconnaissance**

Review of application architecture, identification of entry points, mapping of IPC channels, and dependency inventory.

### **Phase 2: Automated Analysis**

Electronegativity static analysis scan, npm audit for dependency vulnerabilities, and configuration review.

### **Phase 3: Manual Review**

Code review of security-critical components, validation of automated findings, identification of false positives, and business logic security assessment.

### **Phase 4: Remediation Verification**

Implementation of security fixes, re-scan to confirm remediation, and documentation of accepted risks.

## **3.2 Risk Rating Methodology**

|  |  |  |
| --- | --- | --- |
| **Rating** | **CVSS Range** | **Description** |
| Critical | 9.0 - 10.0 | Immediate exploitation possible, severe impact |
| High | 7.0 - 8.9 | Exploitation likely, significant impact |
| Medium | 4.0 - 6.9 | Exploitation possible, moderate impact |
| Low | 0.1 - 3.9 | Exploitation unlikely, minimal impact |
| Informational | 0.0 | Best practice recommendation |

# **4. Findings Detail**

## **4.1 FINDING-001: Missing Navigation Limits**

|  |  |
| --- | --- |
| **Attribute** | **Value** |
| ID | FINDING-001 |
| Title | Missing Navigation Limits (LIMIT\_NAVIGATION\_GLOBAL\_CHECK) |
| Severity | HIGH |
| CVSS Score | 7.5 |
| Status | REMEDIATED |
| CWE | CWE-601: URL Redirection to Untrusted Site |

### **Description**

The application did not implement navigation event handlers to restrict where the BrowserWindow could navigate. This allows potential attackers to redirect the application to malicious external URLs if they can inject content or manipulate links.

### **Attack Vector**

An attacker could exploit this by: (1) Crafting a malicious quote file containing JavaScript that triggers navigation, (2) Exploiting any XSS vulnerability to redirect to a phishing site, or (3) Using protocol handlers to navigate to external resources.

### **Remediation Applied**

// main.js - Navigation limit implementation

mainWindow.webContents.on('will-navigate', (event, navigationUrl) => {

const parsedUrl = new URL(navigationUrl);

if (parsedUrl.protocol !== 'file:') {

event.preventDefault();

console.warn(`[Security] Blocked navigation to: ${navigationUrl}`);

}

});

|  |
| --- |
| **✓ Verification**  Post-remediation scan confirms the finding is resolved. Navigation attempts to external URLs are now blocked and logged. |

## **4.2 FINDING-002: Missing Content Security Policy**

|  |  |
| --- | --- |
| **Attribute** | **Value** |
| ID | FINDING-002 |
| Title | No Content Security Policy (CSP\_GLOBAL\_CHECK) |
| Severity | MEDIUM |
| CVSS Score | 6.1 |
| Status | REMEDIATED |
| CWE | CWE-79: Cross-site Scripting (XSS) |

### **Description**

The application did not implement a Content Security Policy, leaving it vulnerable to cross-site scripting (XSS) attacks and unauthorized resource loading.

### **Remediation Applied**

const CSP\_HEADER = [

"default-src 'self'",

"script-src 'self' 'unsafe-inline' 'unsafe-eval' blob:",

"style-src 'self' 'unsafe-inline'",

"img-src 'self' data: blob:",

"font-src 'self' data:",

"connect-src 'self'",

"worker-src 'self' blob:",

"frame-src 'none'",

"object-src 'none'",

"base-uri 'self'"

].join('; ');

### **CSP Directive Analysis**

|  |  |  |
| --- | --- | --- |
| **Directive** | **Setting** | **Rationale** |
| default-src | 'self' | Restrict all resources to same origin |
| script-src | 'self' 'unsafe-inline' 'unsafe-eval' blob: | Required for React/Babel runtime |
| style-src | 'self' 'unsafe-inline' | Allow inline styles for UI |
| frame-src | 'none' | Block all iframes |
| object-src | 'none' | Block plugins (Flash, Java) |

|  |
| --- |
| **✓ Verification**  CSP headers are now applied to all responses, restricting resource loading to approved sources. |

## **4.3 FINDING-003: Missing Permission Request Handler**

|  |  |
| --- | --- |
| **Attribute** | **Value** |
| ID | FINDING-003 |
| Title | No Permission Request Handler |
| Severity | MEDIUM |
| CVSS Score | 5.3 |
| Status | REMEDIATED |
| CWE | CWE-269: Improper Privilege Management |

### **Remediation Applied**

session.defaultSession.setPermissionRequestHandler((webContents, permission, callback) => {

const allowedPermissions = ['clipboard-read', 'clipboard-write', 'clipboard-sanitized-write'];

if (allowedPermissions.includes(permission)) {

callback(true);

} else {

console.warn(`[Security] Denied permission request: ${permission}`);

callback(false);

}

});

### **Permissions Matrix**

|  |  |  |
| --- | --- | --- |
| **Permission** | **Status** | **Justification** |
| clipboard-read | ALLOW | Required for copy/paste |
| clipboard-write | ALLOW | Required for copy/paste |
| media | DENY | Not needed |
| geolocation | DENY | Not needed |
| notifications | DENY | Not needed |
| fullscreen | DENY | Fixed window size |

## **4.4 FINDING-004: Auxclick Window Opening**

|  |  |
| --- | --- |
| **Attribute** | **Value** |
| ID | FINDING-004 |
| Title | Middle-Click Window Opening (AUXCLICK\_JS\_CHECK) |
| Severity | MEDIUM |
| CVSS Score | 4.3 |
| Status | REMEDIATED |
| CWE | CWE-1021: Improper Restriction of Rendered UI Layers |

### **Remediation Applied**

mainWindow.webContents.setWindowOpenHandler(({ url }) => {

console.warn(`[Security] Blocked new window request for: ${url}`);

return { action: 'deny' };

});

|  |
| --- |
| **✓ Verification**  All new window creation attempts are now blocked and logged. |

## **4.5 FINDING-005: Sandbox Disabled (ACCEPTED RISK)**

|  |  |
| --- | --- |
| **Attribute** | **Value** |
| ID | FINDING-005 |
| Title | Sandbox Disabled (SANDBOX\_JS\_CHECK) |
| Severity | MEDIUM |
| CVSS Score | 5.0 |
| Status | ACCEPTED RISK |
| CWE | CWE-265: Privilege Issues |

### **Business Justification**

The sandbox must be disabled to support: (1) MAC address retrieval for hardware-based license verification, (2) File system access for settings.json persistence, and (3) Native dialog integration for quote file management.

### **Compensating Controls**

|  |  |  |
| --- | --- | --- |
| **Control** | **Status** | **Description** |
| nodeIntegration | Disabled | Prevents direct Node.js access in renderer |
| contextIsolation | Enabled | Isolates preload context from renderer |
| Navigation Limits | Implemented | Blocks external URL navigation |
| CSP | Implemented | Restricts resource loading |
| Permission Handler | Implemented | Blocks unauthorized permissions |

|  |
| --- |
| **⚠ Risk Acceptance**  This risk is accepted with the documented compensating controls. Signed: Psyrcuit LLC Security Team, January 30, 2026 |

## **4.6 FINDING-006: Preload Script Usage**

|  |  |
| --- | --- |
| **Attribute** | **Value** |
| ID | FINDING-006 |
| Title | Preload Script Review (PRELOAD\_JS\_CHECK) |
| Severity | INFORMATIONAL |
| Status | REVIEWED - SECURE |

### **Review Findings**

|  |  |  |
| --- | --- | --- |
| **Criterion** | **Status** | **Notes** |
| Uses contextBridge | ✓ | Proper API isolation |
| No raw Node exposure | ✓ | Only specific functions exposed |
| Input validation | ✓ | Parameters validated before use |
| Limited API surface | ✓ | Purpose-specific functions only |
| No eval() usage | ✓ | No dynamic code execution |

|  |
| --- |
| **✓ Conclusion**  No action required. Implementation follows Electron security best practices. |

# **5. Risk Assessment Matrix**

## **5.1 Pre-Remediation**

|  |  |  |  |
| --- | --- | --- | --- |
| **Finding** | **Likelihood** | **Impact** | **Risk Level** |
| Navigation Limits | Medium | High | HIGH |
| Missing CSP | Medium | Medium | MEDIUM |
| Permission Handler | Low | Medium | MEDIUM |
| Auxclick Windows | Low | Medium | MEDIUM |
| Sandbox Disabled | Low | High | MEDIUM |

## **5.2 Post-Remediation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Finding** | **Likelihood** | **Impact** | **Risk Level** | **Status** |
| Navigation Limits | Very Low | High | LOW | Remediated |
| Missing CSP | Very Low | Medium | LOW | Remediated |
| Permission Handler | Very Low | Medium | LOW | Remediated |
| Auxclick Windows | Very Low | Medium | LOW | Remediated |
| Sandbox Disabled | Very Low | High | LOW | Accepted |

# **6. Remediation Summary**

## **6.1 Changes Made**

|  |  |  |
| --- | --- | --- |
| **File** | **Change Type** | **Description** |
| main.js | Addition | CSP\_HEADER constant (lines 21-32) |
| main.js | Addition | will-navigate handler (lines 128-135) |
| main.js | Addition | setWindowOpenHandler (lines 139-143) |
| main.js | Addition | setPermissionRequestHandler (lines 147-161) |
| main.js | Addition | CSP header injection (lines 165-172) |
| main.js | Modification | Added 'session' to require statement |
| package.json | Removal | Deprecated publisherName property |

## **6.2 Metrics**

|  |  |
| --- | --- |
| **Metric** | **Count** |
| Lines Added | 65 |
| Lines Modified | 2 |
| Lines Removed | 1 |
| Total Files Changed | 2 |

## **6.3 Dependency Updates**

|  |  |  |  |
| --- | --- | --- | --- |
| **Package** | **Before** | **After** | **Reason** |
| electron | 28.1.0 | 40.1.0 | Security patches |
| electron-builder | 24.9.1 | 26.6.0 | Security patches |

# **7. Security Architecture Review**

## **7.1 Security Controls Summary**

|  |  |  |
| --- | --- | --- |
| **Layer** | **Control** | **Status** |
| Process Isolation | contextIsolation: true | ✓ |
| Node Access | nodeIntegration: false | ✓ |
| Navigation | will-navigate handler | ✓ |
| New Windows | setWindowOpenHandler | ✓ |
| Permissions | setPermissionRequestHandler | ✓ |
| Content Policy | CSP headers | ✓ |
| API Exposure | contextBridge only | ✓ |

# **8. Recommendations**

## **8.1 Immediate (Completed)**

✓ Implement navigation limits

✓ Add Content Security Policy

✓ Add permission request handler

✓ Block new window creation

✓ Update dependencies to patched versions

## **8.2 Short-Term (v3.1.0)**

○ Consider pre-compiling React to remove 'unsafe-eval' from CSP

○ Implement code signing for distributed executables

○ Add automated security scanning to CI/CD pipeline

## **8.3 Long-Term**

○ Investigate alternative license verification to enable sandbox

○ Engage third-party penetration testing firm for annual assessment

○ Implement security logging and monitoring

○ Consider Electron Fuses for additional hardening

# **9. Appendix A: Tool Output**

## **9.1 Electronegativity Scan Results (Pre-Remediation)**

$ electronegativity -i . -o security-report.csv

Electronegativity v1.10.2

Scanning: /workspaces/MFC

Files analyzed: 4

Results:

┌────────────────────────────────────────────┬──────────┬────────────┐

│ Check │ Severity │ Location │

├────────────────────────────────────────────┼──────────┼────────────┤

│ LIMIT\_NAVIGATION\_GLOBAL\_CHECK │ HIGH │ Global │

│ CSP\_GLOBAL\_CHECK │ MEDIUM │ Global │

│ PERMISSION\_REQUEST\_HANDLER\_GLOBAL\_CHECK │ MEDIUM │ Global │

│ AUXCLICK\_JS\_CHECK │ MEDIUM │ main.js │

│ SANDBOX\_JS\_CHECK │ MEDIUM │ main.js │

│ PRELOAD\_JS\_CHECK │ MEDIUM │ main.js │

└────────────────────────────────────────────┴──────────┴────────────┘

Total: 7 findings (0 Critical, 1 High, 5 Medium, 1 Informational)

## **9.2 npm Audit Results (Post-Remediation)**

$ npm audit

found 0 vulnerabilities

# **10. Appendix B: Code Changes**

## **10.1 Import Statement Change**

// Before

const { app, BrowserWindow, Menu, ipcMain, dialog, shell, screen } = require('electron');

// After

const { app, BrowserWindow, Menu, ipcMain, dialog, shell, screen, session } = require('electron');

## **10.2 Security Configuration Added**

// ============ Security Configuration ============

const CSP\_HEADER = [

"default-src 'self'",

"script-src 'self' 'unsafe-inline' 'unsafe-eval' blob:",

"style-src 'self' 'unsafe-inline'",

"img-src 'self' data: blob:",

"font-src 'self' data:",

"connect-src 'self'",

"worker-src 'self' blob:",

"frame-src 'none'",

"object-src 'none'",

"base-uri 'self'"

].join('; ');

# **Report Certification**

This report accurately represents the security assessment findings for Metal Finishing Calculator v3.0.0.

**Prepared by: Psyrcuit LLC Security Team**

Date: January 30, 2026

Report Reference: MFC-SEC-2026-001-FULL

Psyrcuit LLC • Tucson, Arizona • security@psyrcuit.com