BearingPoint®

srvc-portal.abc - Web

Advanced Threat Inspection

***Anmerkung:** Hierbei handelt es sich um einen echten Bericht, aus diesem Grund wurden Screenshots in den Ergebnissen geschwärzt und alle Daten die auf den Kunden zurückzuführen könnten entfernt.

Greybox Service Test

V.1.0

Firma

ACME Schulz GmbH
Hauptstraße 1
8010 Graz
Österreich

Datum

2022-07-22

Autoren

Almir Ahmetagić
Georg Lerchbaum
Marcel Schnideritsch

1. Dokumenteigenschaften

Titel	srvc-portal.abc - Web Advanced Threat Inspection
Version	1.0
Autoren	Almir Ahmetagić Georg Lerchbaum Marcel Schnideritsch
Tester	Almir Ahmetagić Georg Lerchbaum Marcel Schnideritsch
Überprüft von	Erlend Depine
Freigegeben von	Erlend Depine
Einstufung	vertraulich

2. Versionskontrolle

Version	Datum	Autoren	Beschreibung
V0.1	2022-07-20	Almir Ahmetagić Georg Lerchbaum Marcel Schnideritsch	Report
V1.0	2022-07-22	Erlend Depine	Review and final version

3. Verteilung

Kopie Nr.	Firma	Name	Datum
1	ACME Schulz GmbH	Max Mustermann	2022-07-24

4. Inhalt

5. Zusammenfassung

Dieses Dokument beschreibt die Ergebnisse der Sicherheitsüberprüfung des srvc-poral.abc Services. Die Sicherheit des Service wurde mittels eines Penetration-Tests gegen eine vom Kunden bereitgestellten Testumgebung evaluiert. Das Ziel war es etwaige Einfallstore für Angreifer zu finden und Softwareprobleme zu dokumentieren, welche einem Angreifer von Vorteil sein könnten. Ebenso sollten die gefundenen Sicherheitsprobleme nach Risiko bewertet werden.

5.1. Arbeitsumfang

Der Test zielte auf das unter "web.srvc-portal.abc" erreichbare System ab. Hierbei wurden auch die damit verbundenen Systeme "api.srvz-portal.abc" und "auth.srvz-portal.abc" evaluiert. Ziel des Tests war es das Webservice zu testen und nicht die darunterliegende Plattform. Um den Test ausführlich durchzuführen, wurden den Testern mehrere Accounts für die Systeme zur Verfügung gestellt.

5.2. Projektziele

Um den Sicherheitszustand des Service bestmöglich zu evaluieren, wurde auf eine möglichst breite Suche von Fehlern gesetzt. Das heißt, es wurden mehrere Möglichkeiten getestet, um dem System Schaden zuzufügen. Gefundene Möglichkeiten wurden ausgenutzt, um einen besseren Einblick für die Risikobewertung zu erlangen. Das Risiko der einzelnen Sicherheitsprobleme wurde nach dem Test basierend auf den Faktoren Wahrscheinlichkeit und Auswirkung bestimmt.

5.3. Annahmen

Während dem Test gingen die Tester von mehreren Angriffsszenarien aus. Unter anderem wurde angenommen, dass Nutzer auch anderen schaden wollen, oder Kunden der Seite und deren Betreibern direkt schaden wollen.

5.4. Zeitplan

Testphase	**Reconnaissance** 	**Pentest** 	**Report** 
Startdatum	2022-07-01	2022-07-05	2022-07-20
Enddatum	2022-07-05	2022-07-20	2022-07-22

5.5. Zusammenfassung Testvorgang

Der Test gliederte sich in mehrere Phasen. Zu Beginn wurden die Funktionalitäten der Webseite analysiert und aufgezeichnet, um sich ein Bild des Verhaltens der Seite zu machen. Dabei wurden auch mögliche risikoreiche Interaktionen notiert, die später genauer betrachtet werden sollten. Die Funktionalität aller Typen von Accounts wurde ebenso erhoben.

Neben Interaktionen mit der Webseite wurde dasselbe auch für die bereitgestellte API gemacht. Auch die Sicherheitseinstellungen und Maßnahmen der Seite wurden betrachtet. Nach einer ausführlichen Analyse wurden die kritischen Funktionen des Systems manuell und mittels Tools getestet. Gefundene Probleme wurden als Risiken aufgenommen und Beispielangriffe als Beweise aufgezeichnet.

Am Ende des Tests wurden alle Probleme in Funktionen oder Einstellungen notiert und deren Risiko bewertet. Diese Auflistung wurde dem Kunden mit diesem Report übergeben. Neben einer Liste von Problemen und Beweisen beinhaltet diese auch Empfehlungen, wie man die Fehler beheben kann.

5.6. Zusammenfassung der Testergebnisse

Bewertung	Notiz	Niedrig	Mittel	Hoch	Kritisch
	3		1	1	2

Die Ergebnisse zeigen, dass in der Entwicklung auf sicherheitsrelevante Einstellungen und Verhalten geachtet wurde. So kann die Seite nur über eine sichere Verbindung verwendet werden, Cookies werden vernünftig verwendet und Eingabefelder scheinen im Backend korrekt vor SQL Injections zu schützen.

Dennoch wurden von den Testern einige Sicherheitslücken entdeckt. So liefert die API sensible Daten an nicht autorisierte Nutzer aus. Ebenso fehlt eine Validierung von eingegebenen oder hochgeladenen Daten. In Kombination mit nicht idealen Sicherheitseinstellungen, erlaubt dieses Problem einem Nutzer den Administrator Status zu erlangen.

5.7. Zusammenfassung der empfohlenen Maßnahmen

Die Ergebnisse zeigen, dass in der Entwicklung auf sicherheitsrelevante Einstellungen und Verhalten geachtet wurde. So kann die Seite nur über eine sichere Verbindung verwendet werden, Cookies werden vernünftig verwendet und Eingabefelder scheinen im Backend korrekt vor SQL Injections zu schützen.

Dennoch wurden von den Testern einige Sicherheitslücken entdeckt. So liefert die API sensible Daten an nicht autorisierte Nutzer aus. Ebenso fehlt eine Validierung von eingegebenen oder hochgeladenen Daten. In Kombination mit nicht idealen Sicherheitseinstellungen, erlaubt dieses Problem einem Nutzer den Administrator Status zu erlangen.

6. Vorgehen

Dieses Kapitel behandelt das Vorgehen während des Tests.

6.1. Reconnaissance

Während der Reconnaissance-Phase wurden folgende Informationen vom Zielsystem eingeholt:

- Welchen Zweck erfüllt das Service?
- Welche Zugangsmöglichkeiten gibt es für das Service?
- Welche verschiedenen Berechtigungsstufen gibt es und wie hängen diese zusammen?
- Welche Funktionalitäten hat die Webseite und wo kann es hier zu Problemen kommen?
- Wie werden User authentifiziert?
- Unterscheidet sich die API Dokumentation mit der realen Funktionalität?
- Welche Risiken würden für den Auftraggeber bei einer Serviceübernahme entstehen?

Die Ergebnisse aus dieser Phase wurden während dem Test immer wieder erneuert und angepasst, je nachdem welche neuen Resultate den Wissensstand der Tester änderten.

6.2. Analyse

In der Analysephase wurde das Verhalten der Webseite und der verfügbaren API genau studiert. Hier wurde sowohl manuell geprüft als auch viele Schritte automatisiert. Zudem wurde ein Sicherheitsscanner verwendet, um offensichtliche und bereits bekannte Probleme im System zu finden.

In der Analyse Phase wurden auch Eingaben auf allgemeine Fehler wie SQL Injections oder XML Injections getestet. Des Weiteren wurden Zugriffsmethoden auf Sicherheitskriterien überprüft, wie Lebensdauer und Verwendung von Cookies und anderen Tokens. Auch Einstellungen, die generell auf sichere Systeme hinweisen, wie CSP oder CSRF Tokens, wurden analysiert und evaluiert.

6.3. Exploitation

In der Exploitation Phase wurden die gefundenen Sicherheitslücken gezielt ausgenutzt. Um die Auswirkung der gefundenen Probleme zu zeigen, wurde ein "Proof of Concept" entworfen. Hierbei wird eine XSS Lücke genutzt, um den JWT Token eines Administrators an einen Angreifer, mit normalen Nutzerrechten, weiterzuleiten. Im Anschluss wird für diesen Angreifer auch ein weiterer Administrator Account angelegt. Dieser Exploit funktioniert, da die Seite eine zu offene Content-Security-Policy hat, keine CSRF Tokens verwendet und JWT Tokens im globalen JavaScript Scope ablegt.

6.4. Risikobewertung

Das Risiko jedes Sicherheitsproblems wird anhand von mehreren Faktoren bewertet. Das Gesamtrisiko für jede Sicherheitslücke wird anhand der folgenden Formel berechnet:

$$\boldsymbol{Risiko = Wahrscheinlichkeit * Auswirkung}$$

		Risiko		
Auswirkung	Hoch	Mittel	Hoch	Kritisch
	Mittel	Niedrig	Mittel	Hoch
	Niedrig	Notiz	Niedrig	Mittel
		Niedrig	Mittel	Hoch
		Wahrscheinlichkeit		

Die Risikobewertung erfolgt in mehreren Schritten:

1. **Risiko benennen**

   Die Tester beschreiben Methoden und Zugriffe, die dem System schaden können. Hierzu werden wirtschaftliche und technische Auswirkungen behandelt.

2. **Bewerten der Wahrscheinlichkeit, dass die Lücke ausgenutzt wird**

   Diese Wahrscheinlichkeit basiert auf mehreren Faktoren

   a. Eigenschaften des Angreifers
      - Können
      - Motiv
      - Möglichkeiten
      - Ressourcen

   b. Eigenschaften der Lücke
      - Wie schwer ist es, die Lücke zu finden?
      - Wie schwer ist es, die Lücke auszunutzen?
      - Ist die Lücke (öffentlich) bekannt?
      - Wie schwierig ist es, zu erkennen, dass die Lücke ausgenutzt wurde (IDS)?

3. **Bewerten der Auswirkungen**

   Es gibt verschiedene Arten von möglichen Auswirkungen.

   a. Technische Auswirkungen
      - Verlust oder Diebstahl von sensiblen Daten
      - Zerstörte Daten
      - Service- oder Systemversagen
      - Kann Datendiebstahl erkannt werden?

   b. Wirtschaftliche Auswirkungen
      - Finanzieller Schaden
      - Image Schaden
      - Gesetzesübertretungen

4. **Bewertung der Risiken anhand der Werte für Wahrscheinlichkeit und Auswirkung**

5. **Anpassen der Ergebnisse anhand von empirischen Werten**

6. **Erstellen von Empfehlungen, wie mit dem jeweiligen Risiko umgegangen werden soll**

Anmerkung: In diesem Fall beachten wir für die Risikobewertung das Szenario „Interner Angreifer“. Die Risiken werden unter der Annahme bewertet, dass ein Angreifer bereits Zugriff auf das interne Netzwerk hat.

7. Detaillierte Ergebnisse

In diesem Abschnitt werden die Ergebnisse der Tests der Webapplikation detailliert beschrieben.

7.1. Detaillierte Systeminformationen

System Adresse	IP-Adresse	System Typ	Information
web.srvc-portal.abc	1.23.456.7, 1.23.456.8	Web Portal	Generelle User Interaktion
api.srvc-portal.abc	1.23.456.7, 1.23.456.8	Rest API	Interaktion für Apps, Oder Webseite
auth.srvc-portal.abc	1.23.456.7, 1.23.456.8		

Software Name	Version	Anmerkung
Nginx	1.14.2	
PHP	7.0.29	Sollte ein Update bekommen, da offiziell End Of Life.

7.2. Stored Cross Site Scripting in mehreren Inputfeldern

Wahrscheinlichkeit	Auswirkung	Risiko
Hoch	Hoch	Kritisch

7.2.1. Analyse

Während der Exploitation-Phase wurden in fast sämtlichen textlichen Eingabefeldern XSS-Probleme vorgefunden. Diese Lücken konnten dazu verwendet werden, beliebige HTML und JavaScript Funktionalitäten einzufügen. Dies kann zu Problemen in der Funktion der Seite führen. In Kombination mit Problemen in der Content-Security-Policy können auch Daten wie Admin API-Tokens an Dritte weitergegeben werden. Die Kombination mit anderen Lücken macht das Risiko dieses Problems kritisch.

7.2.2. Empfehlung

Die Eingabe von Textfeldern sollte limitiert und Felder entweder richtig behandelt oder die zu verwendeten Zeichen einschränkt werden. JavaScript Variablen sollten lokal verwendet werden, um einen Zugriff von anderen Funktionen zu limitieren. Dies würde ein Abhandenkommen des JWT Token verhindern.

7.3. Bilderupload ermöglicht das Hochladen von beliebigen Filetypen

Wahrscheinlichkeit	Auswirkung	Risiko
Hoch	Hoch	Kritisch

7.3.1. Analyse

Die Funktionalität, Bilder von Usern hochzuladen beinhaltet nur eine Überprüfung der Fileextension in JavaScript. Ein Angreifer kann diesen Code sehr leicht ändern und damit den Check umgehen. Dadurch können beliebige Files hochgeladen werden (PDF, HTML, etc.). Diese Files könnten später in Fishing Angriffen oder in erweiterten XSS Angriffen genutzt werden, um einen einfachen User zu einem Administrator zu machen. Die Kombination mit anderen Lücken macht das Risiko dieses Problems kritisch.

7.3.2. Empfehlung

Der Content-Typ beim Hochladen sollte auf "image/* " überprüft werden, um das Anzeigen und Rendern von hochgeladenen HTML Files zu verhindern.

7.4. API Endpunkte beinhalten keine Sicherheitsüberprüfung

Wahrscheinlichkeit	Auswirkung	Risiko
Hoch	Mittel	Hoch

7.4.1. Analyse

Diverse API Endpunkte, die unter https://api.srvc-portal.abc/api-docs als geschützt oder nur für besondere Nutzer zugänglich sind, werden nicht korrekt behandelt. Endpunkte wie "/customers" geben ihre Information ohne Authentifizierung preis. Hierbei werden Datenschutzrelevante Daten an Unbekannte weitergegeben.

7.4.2. Empfehlung

Korrekte Überprüfung der Zugriffsberechtigungen für alle API-Endpunkte.

7.5. Content Security Policy beinhaltet zu offene Richtlinien

Wahrscheinlichkeit	Auswirkung	Risiko
Mittel	Mittel	Mittel

7.5.1. Analyse

Während des Testens wurde festgestellt, dass die Content Security Policy (CSP) einem potenziellen Angreifer viel Freiheiten lässt. Beschränkungen wie zum Beispiel "https://*.amazonaws.com" sind wenig aussagekräftig, da diese auf jede AWS Instanz zutreffen und Jeder einen Server mit einer Adresse in diesem Format hosten kann. Die Kombination mit anderen Lücken macht das Risiko dieses Problems mittel.

7.5.2. Empfehlung

Überarbeiten der Content Security Policy und die Limitierungen auf ganze Domains setzen.

7.6. Keine Limitierung der Login Versuche

Wahrscheinlichkeit	Auswirkung	Risiko
Niedrig	Niedrig	Notiz

7.6.1. Analyse

Die Anmeldeversuche beim Login sind nicht limitiert und können beliebig oft und in sehr kurzer Zeit erfolgen. In Kombination mit einer wenig restriktiven Passwort Policy ermöglicht dies, Zugänge von bekannten Nutzern durch Brute Force zu erraten.

7.6.2. Empfehlung

Es wird empfohlen ein Timeout bei Versuchen pro IP pro Zeiteinheit einzuführen, um einen möglichen Angriff zu verlangsamen. Eine generelle Limitierung würde es einem Angreifer ermöglichen einen Nutzer vom Service auszusperren.

7.7. Fehlende Passwort Policy

Wahrscheinlichkeit	Auswirkung	Risiko
Niedrig	Niedrig	Notiz

7.7.1. Analyse

Beim Erstellen eines neuen Users gibt es keine Vorschriften wie ein Passwort auszusehen hat. Dies verleitet zur Verwendung von unsicheren und kurzen Passwörtern. Im Zusammenhang mit der fehlenden Limitierung von Anmeldeversuchen erhöht dies die Erfolgschance von Brute Force Angriffen.

7.7.2. Empfehlung

Einrichtung einer Passwort Policy von mindestens 12 Zeichen, davon eine Ziffer, ein Sonderzeichen und Klein-/Großbuchstaben.

7.8. Fehlende CSRF Tokens

Wahrscheinlichkeit	Auswirkung	Risiko
Niedrig	Niedrig	Notiz

7.8.1. Analyse

Da die Seite keine "cross-site request forgery"-Tokens hat, kann ein Angreifer Ab- und Anfragen an die Webseite senden. Dies kann in Kombination mit XSS oder HTML Injection Lücken einen Angriff vereinfachen.

7.8.2. Empfehlung

Einrichtung von CSRF Token für alle Form Felder.

7.9. Proof of Concept Exploit Kette

Unser Exploit zielt darauf ab, einem normalen Nutzer Administratorrechte zu verschaffen. Für diesen Exploit wird eine "Stored XSS" im Eingabekommentar ausgenutzt, um einen Admin dazu zu bringen, einen zusätzlichen Nutzer mit Administratorrechten anzulegen. Über einen Request an eine dritte Seite werden dem Angreifer dann Account Details mitgeteilt und zusätzlich der JWT Token des Admins zugesendet. Der Exploit nutzt dafür die nicht ideale Content Policy der Seite, die globale JWT Variable und die fehlenden CSRF Tokens aus. Die folgenden Schritte werden für den Exploit ausgeführt, die verwendeten Skripts befinden sich im Appendix:

- Der Angreifer legt einen neuen Eintrag an und nutzt die Kommentarfunktion um den HTML
- Code aus "injection.html" in der Seite einzubinden. Dieser code wird nun für alle User in der /entries Seite eingebunden. Im JavaScript ist ein Check eingebaut, der dafür sorgt, dass der Code nur unter einem Admin ausgeführt wird.
- Der Angreifer betreibt einen Server auf einer Instanz bei Amazon, um eine *amazonaws.com Domain zu erhalten. Dort läuft das "catch_jwt_login.php" Skript. Wenn ein Admin nun entweder den Eintrag des Kunden direkt anschaut oder der Eintrag auf der /entries Seite angezeigt wird, wird der Schadcode ausgeführt.
- Im Schadcode wird der JWT Token zuerst überprüft, dann ein Objekt vom Server des Angreifers geladen, wobei der Request hier einen Userlogin und den JWT beinhaltet.
- Danach wird ein HTML Form an /admins/new gesendet, welches den neuen Admin anlegt.
  5. Der Angreifer kann sich danach als Admin bei der Webseite anmelden.

Das Skript kann nach Belieben geändert werden, zum Beispiel, um nach dem Request automatisch alle Spuren im Eintrag etc. zu verwischen.

Auf Wunsch kann dem Kunden eine Videoaufzeichnung des durchgeführten Angriffs zugesendet werden.

8. Verwendete Software

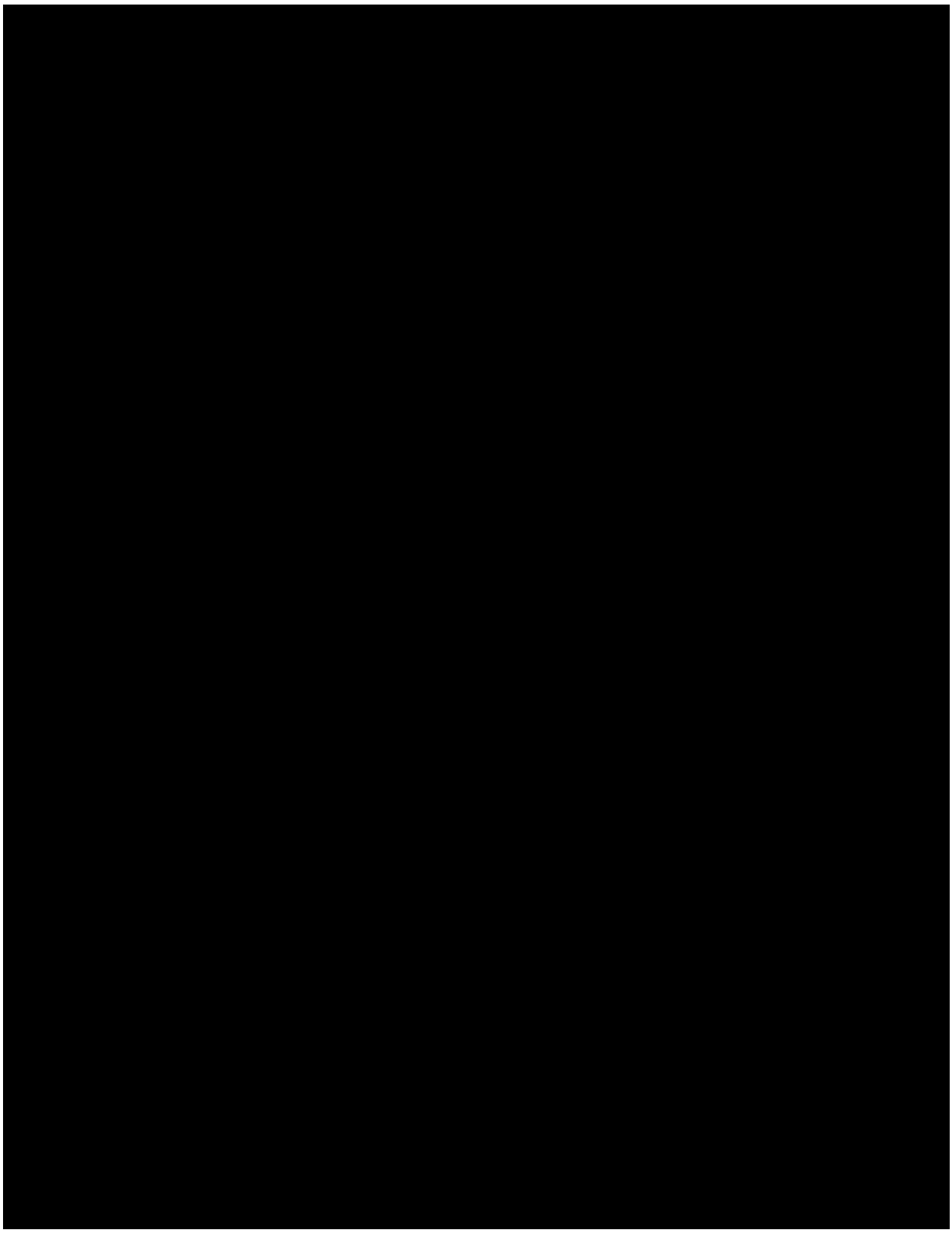
Software	Zweck	Link
nmap	Netzwerkscan	https://nmap.org
Burp Suite	Netzwerk Proxy	https://portswigger.net
THC Hydra	Passwort Brute Force	https://sectools.org/tool/hydra/
dirsearch	Web Pfad Suche	https://github.com/maurosoria/dirsearch
Nessus	Sicherheitsscanner	https://www.tenable.com/products/nessus
SQLmap	SQL-Injection Finder	http://sqlmap.org/

9. Anhang

9.1. Catch jwt login

catch_jwt_login.php

9.2. Injection.html



BearingPoint®

srvc-portal.abc - Web Advanced Threat Inspection