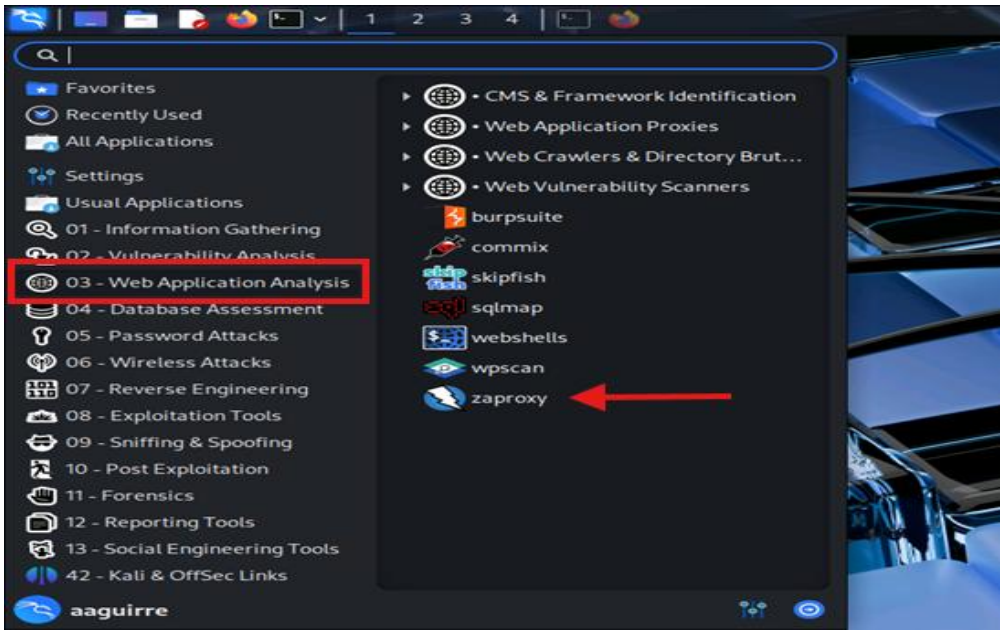
POLICIA FEDERAL ARGENTINA

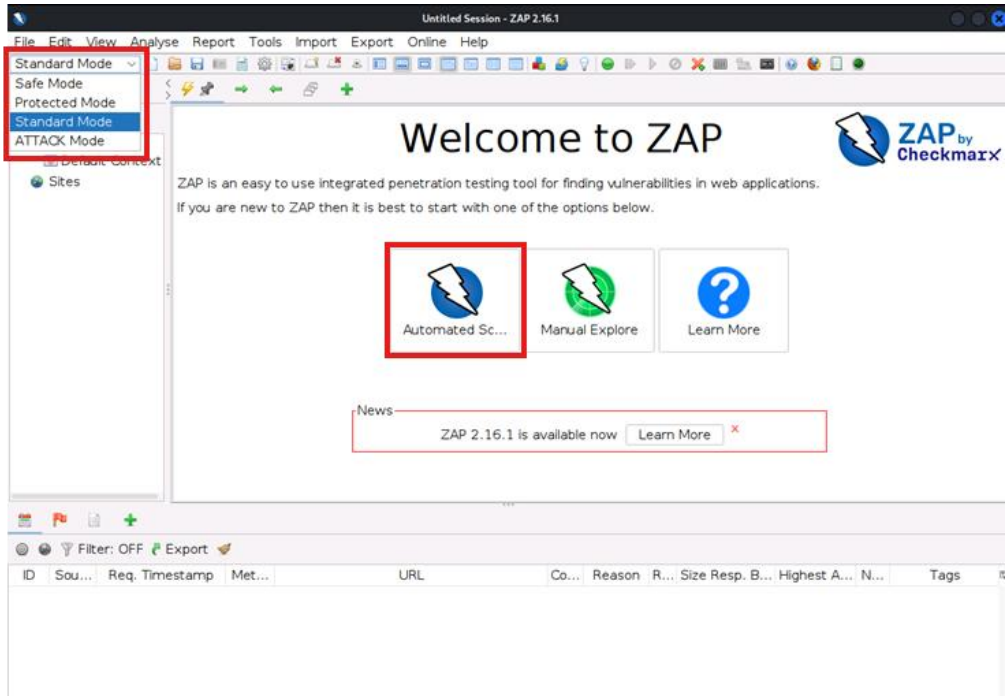
Procedimiento de PENTEST

Superintendencia FEDERAL DE TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIONES

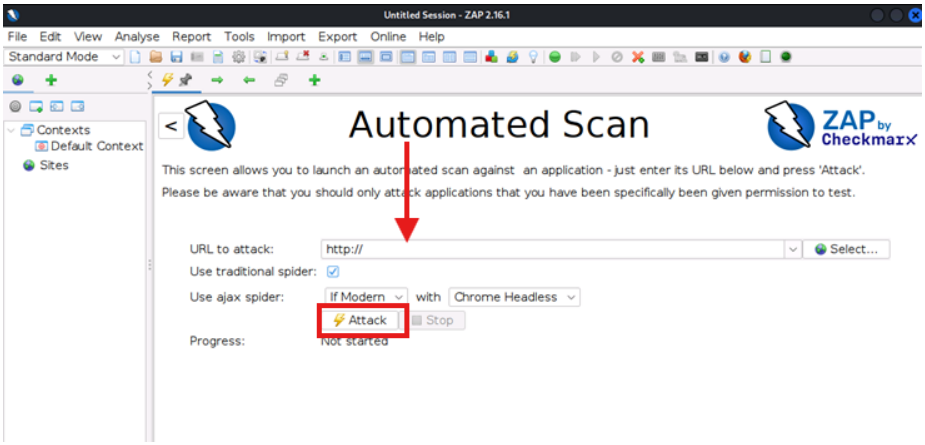
1. Ingresar en la maquna virtual ya sea desde el vmware o mediente RDP. Una vez dentro deberan ingresar en la categoria "**Web Application Analysis**" y acceder a la herramienta con la cual realizaremos el Pentest "**zaproxy**"

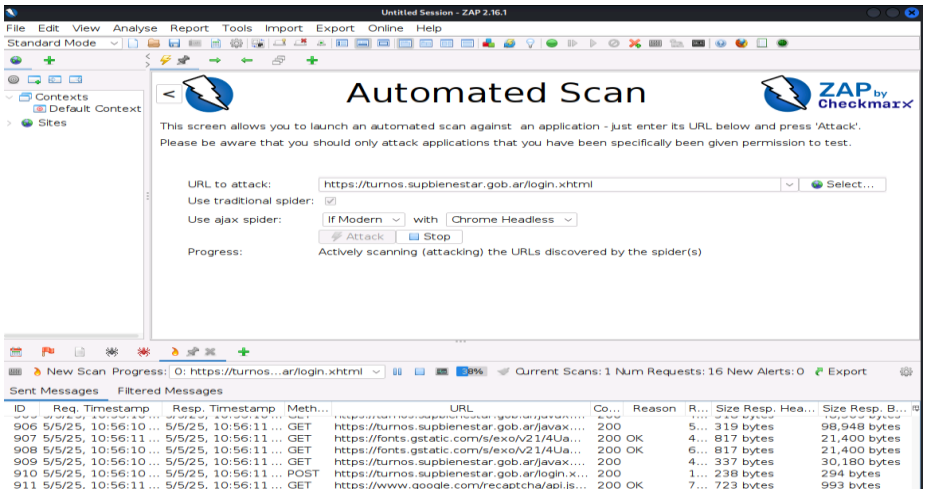


2. Una vez dentro de la herramienta podemos seleccionar el tipo de escaneo que se realizara, el cual deberá estar acorde con los lineamientos aceptados en la autorización de Pentest.

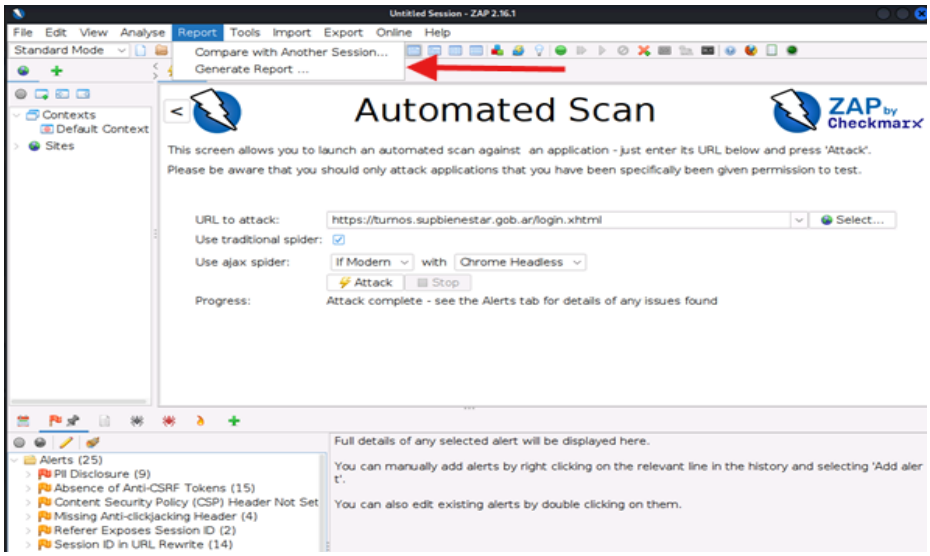


3. Para comenzar acceder en “**Automated Scan**”, el cual les solicitara que proporcionen el **sitio** a escanear. Una vez ingresado selecciones “**Attack**”.



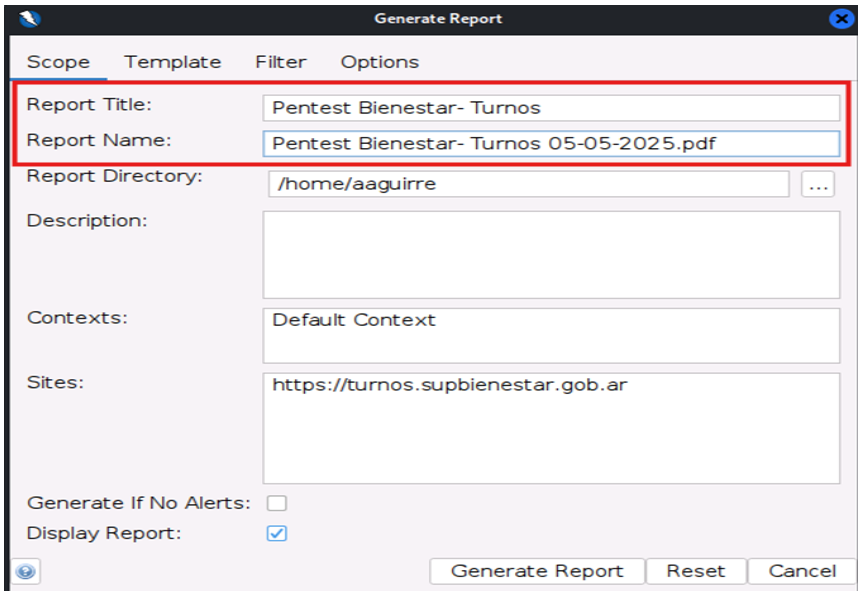


4. Terminado el escaneo deberán dirigirse a la ventana de “**Report**”, **“Generate Report”**

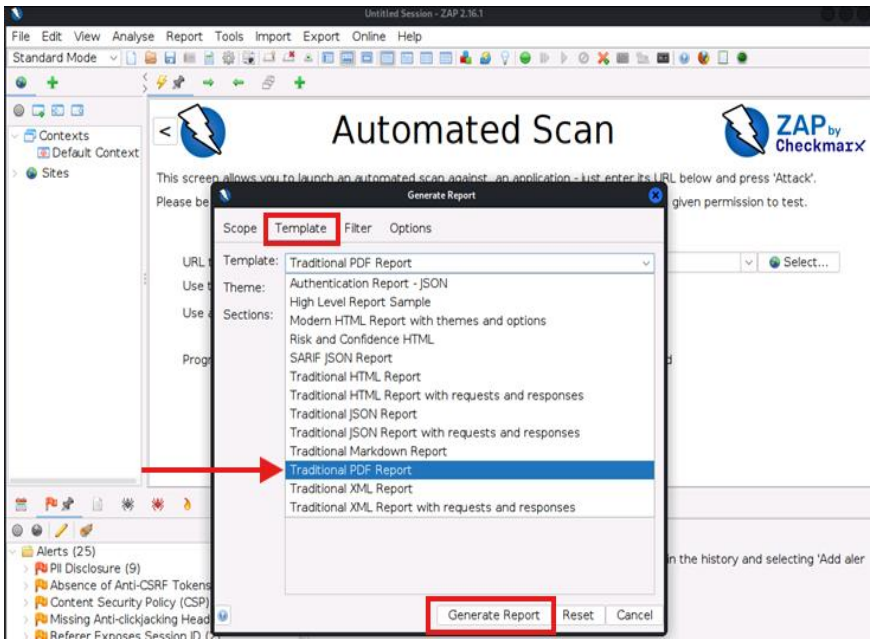


5. Para mantener un lineamiento deberán completar los siguientes campos:

- **Report Title**: con la organización/ dependencia/servicio, que se esté escaneando.
- **Report Name**: Deberá coincidir con el "Report Title", pero al final se le agregara la fecha en la cual se realizó el escaneo, con la finalidad de que sea un reporte con integridad.



6. En la ventana de "Template", tendrán diversos diseños de reportes, en este caso siempre usaremos el "Traditional PDF Report"



7. El reporte les brindara un resumen de las vulnerabilidades detectadas las mismas estarán categorizadas por criticidad

Summary of Alerts

Risk Level	Number of Alerts
High	1
Medium	5
Low	6
Informational	13

Alerts

Name	Risk Level	Number of Instances
PII Disclosure	High	9
Absence of Anti-CSRF Tokens	Medium	15
Content Security Policy (CSP) Header Not Set	Medium	6
Missing Anti-clickjacking Header	Medium	4
Referer Exposes Session ID	Medium	2
Session ID in URL Rewrite	Medium	14
Application Error Disclosure	Low	1
Cookie Without Secure Flag	Low	1

8. Campos de relevancia:

- Descripción: Resumen de la vulnerabilidad detectada, como así también el impacto.
- URL / Evidencia: Dirección en donde se encontró la falta del Anti-CSRF con su evidencia.
- Solución: **RECOMENDACIONES** para su remediación

Medium	Absence of Anti-CSRF Tokens
Description	No Anti-CSRF tokens were found in a HTML submission form. A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL /form actions in a repeatable way. The nature of the attack is that CSRF exploits the trust that a web site has for a user. By contrast, cross-site scripting (XSS) exploits the trust that a user has for a web site. Like XSS, CSRF attacks are not necessarily cross-site, but they can be. Cross-site request forgery is also known as CSRF, XSRF, one-click attack, session riding, confused deputy, and sea surf. CSRF attacks are effective in a number of situations, including: * The victim has an active session on the target site. * The victim is authenticated via HTTP auth on the target site. * The victim is on the same local network as the target site. CSRF has primarily been used to perform an action against a target site using the victim's privileges, but recent techniques have been discovered to disclose information by gaining

	access to the response. The risk of information disclosure is dramatically increased when the target site is vulnerable to XSS, because XSS can be used as a platform for CSRF, allowing the attack to operate within the bounds of the same-origin policy.
URL	https://turnos.supbienestar.gob.ar/login.xhtml
Method	GET
Attack	
Evidence	<form id="j_idt7" name="j_idt7" method="post" action="/login.xhtml;jsessionid=903FB83CA2340909AC670A116B7B66EE" enctype="application/x-www-form-urlencoded">
Other Info	No known Anti-CSRF token [anticsrf, CSRFToken, __RequestVerificationToken, csrfmiddlewaretoken, authenticity_token, OWASP_CSRFTOKEN, anoncsrf, csrf_token, _csrf, _csrfSecret, __csrf_magic, CSRF, _token, _csrf_token, _csrfToken] was found in the following HTML form: [Form 1: "j_id1:javax.faces.ViewState:0" "j_idt7"].

Solution	Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid. For example, use anti-CSRF packages such as the OWASP CSRFGuard. Phase: Implementation Ensure that your application is free of cross-site scripting issues, because most CSRF defenses can be bypassed using attacker-controlled script. Phase: Architecture and Design Generate a unique nonce for each form, place the nonce into the form, and verify the nonce upon receipt of the form. Be sure that the nonce is not predictable (CWE-330). Note that this can be bypassed using XSS. Identify especially dangerous operations. When the user performs a dangerous operation, send a separate confirmation request to ensure that the user intended to perform that operation. Note that this can be bypassed using XSS. Use the ESAPI Session Management control. This control includes a component for CSRF. Do not use the GET method for any request that triggers a state change. Phase: Implementation Check the HTTP Referer header to see if the request originated from an expected page. This could break legitimate functionality, because users or proxies may have disabled sending the Referer for privacy reasons.
Reference	https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html https://cwe.mitre.org/data/definitions/352.html
CWE Id	352
WASC Id	9
Plugin Id	10202