

### About **CODEPROFILER**

CodeProfiler is the 1<sup>st</sup> product worldwide that provides automated security testing for ABAP and BSP applications. Its database contains patterns of the most important insecure coding practices for ABAP.









### Use Cases

CodeProfiler is the tool of choice for establishing a baseline security level for all ABAP-based business applications. Typical use cases are:

- Your company wants to determine whether authority checks are in place when critical business logic is executed.
- Your company wants to check if a supplier built a back door into the code.
- You want to check, if there are any security defects in your ABAP coding that endanger your business assets or violate compliance.

### Test Groups (Excerpt)

Our test groups contain various test patterns that test for typical security defects in ABAP code.

- **Missing Authority Checks**   
ABAP can execute business transactions without privileges. Therefore, whenever ABAP programs call functionality that requires certain privileges to run, an authority check should be made *programmatically*. Otherwise users might get access to restricted functionality.
- **Dangerous ABAP commands**  
These test patterns check if there are any commands used in an ABAP program that could pose a security threat. Examples are access to files and low-level system commands.
- **Backdoors**   
There are several ways to include backdoors in ABAP programs. They allow malicious developers to secretly access extra-functionality by feeding certain triggers to the program.
- **Hard-coded user credentials**   
These test patterns check if there are any hard-coded user credentials in the code.
- **Generic Operations**   
Sometimes developers write code in a way that it can be used for a number of different use cases. This flexibility often results in vulnerabilities when malicious users discover unforeseen use cases nobody expected.
- **Command execution**   
In some instances, ABAP code can be *generated* and executed at runtime. These test patterns check, if such risky practices are used and if they are exploitable.
- **SQL Injection**   
This coding defect allows malicious users to manipulate OSQL statements. This can result in information disclosure and manipulation of arbitrary data in the SAP database.
- **Cross Site Scripting (XSS)**   
XSS is a top risk in every Web application and can lead to identity theft. These test patterns check if BSP applications are vulnerable to XSS.
- **Forceful Browsing**   
In a Web application context, malicious users can bypass UI restrictions such as disabled buttons through Forceful Browsing. These patterns check if such attacks are possible in BSP applications.



This icon indicates vulnerabilities that in most cases also violate regulatory compliance.

