Version 9.6.1 - Queries Release Notes

New Queries:

Language	Group	Name	CWE
Dart	Dart_Mobile_Medium_Threat	Absolute_Path_Traversal	36
Dart	Dart_Mobile_Medium_Threat	Insecure_WebSocket_Connection	319
Go	Go_Medium_Threat	Insecure_Value_of_the_SameSite_Cookie_Attribute_in_Code	1275
Go	Go_Medium_Threat	Trust_Proxy_On	348
Go	Go_Medium_Threat	Unsafe_Object_Binding	0
Lua	Lua_Best_Coding_Practice	Use_of_Evil_Regex	0
Lua	Lua_Best_Coding_Practice	Use_of_Native_Language	695
Lua	Lua_High_Risk	Deserialization_of_Untrusted_Data	74
Lua	Lua_Low_Visibility	Heap_Inspection	244
Lua	Lua_Low_Visibility	Log_Forging	117
Lua	Lua_Low_Visibility	Missing_Content_Security_Policy	346
Lua	Lua_Low_Visibility	Missing_HSTS_Header	346
Lua	Lua_Low_Visibility	PCI_Data_Exposure	200
Lua	Lua_Low_Visibility	PCI_Data_Exposure_in_Error_Messages	200
Lua	Lua_Low_Visibility	PCI_Data_Exposure_in_Files	200
Lua	Lua_Low_Visibility	PCI_Data_Exposure_in_JWT	200
Lua	Lua_Low_Visibility	PCI_Data_Exposure_in_Logs	200
Lua	Lua_Low_Visibility	PCI_Data_Exposure_in_URL	200
Lua	Lua_Low_Visibility	Permissive_Content_Security_Policy	346
Lua	Lua_Low_Visibility	Privacy_Violation_in_Error_Messages	200
Lua	Lua_Low_Visibility	Privacy_Violation_in_Files	200
Lua	Lua_Low_Visibility	Privacy_Violation_in_Logs	200
Lua	Lua_Low_Visibility	Privacy_Violation_in_URL	200
Lua	Lua_Low_Visibility	Secret_Leak_in_Error_Messages	200
Lua	Lua_Low_Visibility	Secret_Leak_in_Files	200
Lua	Lua_Low_Visibility	Secret_Leak_in_Logs	200
Lua	Lua_Low_Visibility	Secret_Leak_in_URL	200
Lua	Lua_Low_Visibility	Server_Information_Exposure	200
Lua	Lua_Low_Visibility	Server_Information_Exposure_via_Misconfiguration	200
Lua	Lua_Low_Visibility	Use_Of_Hardcoded_Password_In_Config	260
Lua	Lua_Medium_Threat	CSRF	352
Lua	Lua_Medium_Threat	Excessive_Data_Exposure	201
Lua	Lua_Medium_Threat	Hashing_Length_Extension_Attack	326
Lua	Lua_Medium_Threat	Secret_Leak	200
Lua	Lua_Medium_Threat	Secret_Leak_in_JWT	200
PHP	PHP_Best_Coding_Practice	Outdated_Encryption_Algorithm	326
PHP	Php_Best_Coding_Practice	Outdated_Hashing_Function	328
PHP	Php_Best_Coding_Practice	Use_of_Evil_Regex	400
PHP	PHP_Low_Visibility	Command_Argument_Injection	88
PHP	PHP_Low_Visibility	Comparison_Timing_Attack	208
PHP	PHP_Low_Visibility	Cookie_Overly_Broad_Path	539
PHP	PHP_Low_Visibility	Cookie_Overly_Broad_Path_In_Config	539
PHP	PHP_Low_Visibility	Error_Messages_Misconfiguration	209
PHP	PHP_Low_Visibility	Missing_Framing_Policy	1021

Language	Group	Name	CWE
PHP	PHP_Low_Visibility	Stored_Command_Argument_Injection	88

Changed Queries:

Language	Group	Name	CWE	Changed Fields
CPP	CPP_Best_Coding_Practice	Buffer_Size_Literal_Condition	118	**Source** has changed
CPP	CPP_Buffer_Overflow	Buffer_Improper_Index_Access	129	**Source** has changed
CPP	CPP_Buffer_Overflow	Buffer_Overflow_LongString	120	**Source** has changed
CPP	CPP_Buffer_Overflow	Buffer_Overflow_Wrong_Buffer_Size	131	**Source** has changed
CPP	CPP_Buffer_Overflow	Improper_Null_Termination	170	**Source** has changed
CPP	CPP_Medium_Threat	Memory_Leak	401	**Source** has changed
CPP	CPP_MISRA_C_2012	R03_X_Comments	0	**Source** has changed
CSharp	CSharp_APISecurity	CSharp_WebApi_GetApiList	0	**Source** has changed
CSharp	CSharp_High_Risk	Deserialization_of_Untrusted_Data_MSMQ	502	**Source** has changed
CSharp	CSharp_High_Risk	Reflected_XSS_All_Clients	79	**Source** has changed
CSharp	CSharp_High_Risk	Stored_XSS	79	**Source** has changed
CSharp	CSharp_Medium_Threat	Unsafe_Object_Binding	915	**Source** has changed
Dart	Dart_Mobile_Best_Coding_Practice	WebView_Cache_Information_Leak	0	**Source** has changed
Dart	Dart_Mobile_High_Risk	Unencrypted_Sensitive_Information_in_Publicly_Accessible_Cloud_Storage	922	**Source** has changed
Dart	Dart_Mobile_High_Risk	Unsafe_Reflection	470	**Source** has changed
Dart	Dart_Mobile_Low_Visibility	Encrypted_Sensitive_Information_in_Publicly_Accessible_Cloud_Storage	922	**Source** has changed
Dart	Dart_Mobile_Low_Visibility	Unencrypted_Sensitive_Information_in_Internal_Storage	922	**Source** has changed
Dart	Dart_Mobile_Low_Visibility	Unencrypted_Sensitive_Information_in_Temporary_File	377	**Source** has changed
Dart	Dart_Mobile_Low_Visibility	User_Information_in_Publicly_Accessible_Storage	922	**Source** has changed
Dart	Dart_Mobile_Medium_Threat	Improper_Certificate_Validation	295	**Source** has changed
Dart	Dart_Mobile_Medium_Threat	Information_Exposure_Through_Query_String	598	**Source** has changed
Dart	Dart_Mobile_Medium_Threat	Third_Party_Keyboards_On_Sensitive_Field	0	**Source** has changed
Go	Go_Low_Visibility	Deprecated_API	477	**Source** has changed
Go	Go_Low_Visibility	Open_Redirect	601	**Source** has changed
Go	Go_Low_Visibility	Race_Condition_In_Cross_Functionality	362	**Source** has changed
Go	Go_Medium_Threat	Cleartext_Transmission_Of_Sensitive_Information	319	**Source** has changed
Go	Go_Medium_Threat	Reflected_Absolute_Path_Traversal	36	**Source** has changed
Go	Go_Medium_Threat	Reflected_Relative_Path_Traversal	23	**Source** has changed
Java	Java_Android	Client_Side_Injection	89	**Source** has changed
Java	Java_Android	Client_Side_ReDoS	400	**Source** has changed
Java	Java_Android	Copy_Paste_Buffer_Caching	922	**Source** has changed
Java	Java_Android	Failure_To_Implement_Least_Privilege	250	**Source** has changed
Java	Java_Android	Implicit_Intent_With_Read_Write_Permissions	668	**Source** has changed
Java	Java_Android	Improper_Verification_Of_Intent_By_Broadcast_Receiver	925	**Source** has changed
Java	Java_Android	Information_Leak_Through_Response_Caching	524	**Source** has changed
Java	Java_Android	Insecure_Data_Storage	312	**Source** has changed
Java	Java_Android	Insecure_Data_Storage_Usage	312	**Source** has changed
Java	Java_Android	Insecure_WebView_Usage	829	**Source** has changed
Java	Java_Android	Insufficient_Application_Layer_Protect	311	**Source** has changed
Java	Java_Android	Insufficient_Sensitive_Application_Layer	319	**Source** has changed
Java	Java_Android	Keyboard_Cache_Information_Leak	524	**Source** has changed
Java	Java_Android	Non_Encrypted_Data_Storage	311	**Source** has changed

Language	Group	Name	CWE	Changed Fields
Java	Java_Android	No_Installer_Verification_Implemented	829	**Source** has changed
Java	Java_Android	Passing_Non_Encrypted_Data_Between_Activities	319	**Source** has changed
Java	Java_Android	Poor_Authorization_and_Authentication	287	**Source** has changed
Java	Java_Android	Side_Channel_Data_Leakage	200	**Source** has changed
Java	Java_Android	Use_Of_Implicit_Intent_For_Sensitive_Communication	927	**Source** has changed
Java	Java_Android	Use_of_Native_Language	695	**Source** has changed
Java	Java_Android	Use_of_WebView_AddJavascriptInterface	749	**Source** has changed
Java	Java_Android	Weak_Encryption	326	**Source** has changed
Java	Java_Android	WebView_Cache_Information_Leak	524	**Source** has changed
Java	Java_APISecurity	Java_WebApi_GetApiList	0	**Source** has changed
Java	Java_AWS_Lambda	AWS_Credentials_Leak	200	**Source** has changed
Java	Java_AWS_Lambda	Hardcoded_AWS_Credentials	798	**Source** has changed
Java	Java_AWS_Lambda	Permission_Manipulation_in_S3	285	**Source** has changed
Java	Java_AWS_Lambda	Race_Condition_Global_Scope	1108	**Source** has changed
Java	Java_Best_Coding_Practice	Access_Specifier_Manipulation	284	**Source** has changed
Java	Java_Best_Coding_Practice	Call_to_Thread_run	572	**Source** has changed
Java	Java_Best_Coding_Practice	Catch_NullPointerException	395	**Source** has changed
Java	Java_Best_Coding_Practice	clone_Method_Without_super_clone	580	**Source** has changed
Java	Java_Best_Coding_Practice	Comparison_of_Classes_By_Name	486	**Source** has changed
Java	Java_Best_Coding_Practice	Critical_Public_Variable_Without_Final_Modifier	493	**Source** has changed
Java	Java_Best_Coding_Practice	Dead_Code	561	**Source** has changed
Java	Java_Best_Coding_Practice	Declaration_of_Throws_for_Generic_Exception	397	**Source** has changed
Java	Java_Best_Coding_Practice	Detection_of_Error_Condition_Without_Action	390	**Source** has changed
Java	Java_Best_Coding_Practice	Direct_Use_of_Sockets	246	**Source** has changed
Java	Java_Best_Coding_Practice	Dynamic_File_Inclusion	829	**Source** has changed
Java	Java_Best_Coding_Practice	Dynamic_Set_Of_Null_SecurityManager	274	**Source** has changed
Java	Java_Best_Coding_Practice	Dynamic_SQL_Queries	89	**Source** has changed
Java	Java_Best_Coding_Practice	Empty_Synchronized_Block	585	**Source** has changed
Java	Java_Best_Coding_Practice	ESAPI_Banned_API	676	**Source** has changed
Java	Java_Best_Coding_Practice	Explicit_Call_to_Finalize	586	**Source** has changed
Java	Java_Best_Coding_Practice	Exposure_of_Resource_to_Wrong_Sphere	493	**Source** has changed
Java	Java_Best_Coding_Practice	finalize_Method_Declared_Public	583	**Source** has changed
Java	Java_Best_Coding_Practice	finalize_Method_Without_super_finalize	568	**Source** has changed
Java	Java_Best_Coding_Practice	GOTO_Statement	699	**Source** has changed
Java	Java_Best_Coding_Practice	Incorrect_Conversion_between_Numeric_Types	681	**Source** has changed
Java	Java_Best_Coding_Practice	Missing_XML_Validation	112	**Source** has changed
Java	Java_Best_Coding_Practice	Pages_Without_Global_Error_Handler	544	**Source** has changed
Java	Java_Best_Coding_Practice	Portability_Flaw_In_File_Separator	474	**Source** has changed
Java	Java_Best_Coding_Practice	Potentially_Serializable_Class_With_Sensitive_Data	499	**Source** has changed
Java	Java_Best_Coding_Practice	Redirect_Without_Exit	698	**Source** has changed
Java	Java_Best_Coding_Practice	Reliance_On_Untrusted_Inputs_In_Security_Decision	807	**Source** has changed
Java	Java_Best_Coding_Practice	Uncontrolled_Recursion	674	**Source** has changed
Java	Java_Best_Coding_Practice	Undocumented_API	1059	**Source** has changed
Java	Java_Best_Coding_Practice	Unused_Variable	563	**Source** has changed
Java	Java_Best_Coding_Practice	Use_of_Inner_Class_Containing_Sensitive_Data	492	**Source** has changed
Java	Java_Best_Coding_Practice	Use_of_Obsolete_Functions	477	**Source** has changed
Java	Java_Best_Coding_Practice	Use_of_System_Output_Stream	398	**Source** has changed

Language	Group	Name	CWE	Changed Fields
Java	Java_GWT	GWT_DOM_XSS	79	**Source** has changed
Java	Java_GWT	GWT_Reflected_XSS	79	**Source** has changed
Java	Java_GWT	JSON_Hijacking	352	**Source** has changed
Java	Java_High_Risk	JSF_Local_File_Inclusion	98	**Source** has changed
Java	Java_High_Risk	Second_Order_SQL_Injection	89	**Source** has changed
Java	Java_High_Risk	Stored_XSS	79	**Source** has changed
Java	Java_Low_Visibility	Authorization_Bypass_Through_User_Controlled_SQL_PrimaryKey	566	**Source** has changed
Java	Java_Low_Visibility	Channel_Accessible_by_NonEndpoint	300	**Source** has changed
Java	Java_Low_Visibility	Cleansing_Canonicalization_and_Comparison_Errors	171	**Source** has changed
Java	Java_Low_Visibility	Collapse_of_Data_into_Unsafe_Value	182	**Source** has changed
Java	Java_Low_Visibility	Cookie_Overly_Broad_Path	539	**Source** has changed
Java	Java_Low_Visibility	Creation_of_Temp_File_in_Dir_with_Incorrect_Permissions	379	**Source** has changed
Java	Java_Low_Visibility	Creation_of_Temp_File_With_Insecure_Permissions	378	**Source** has changed
Java	Java_Low_Visibility	Divide_By_Zero	369	**Source** has changed
Java	Java_Low_Visibility	ESAPI_Same_Password_Repeats_Twice	521	**Source** has changed
Java	Java_Low_Visibility	Exposure_of_System_Data	497	**Source** has changed
Java	Java_Low_Visibility	File_Permissions_World_Readable	732	**Source** has changed
Java	Java_Low_Visibility	Heap_Inspection	244	**Source** has changed
Java	Java_Low_Visibility	Improper_Resource_Access_Authorization	285	**Source** has changed
Java	Java_Low_Visibility	Improper_Resource_Locking	413	**Source** has changed
Java	Java_Low_Visibility	Improper_Resource_Shutdown_or_Release	404	**Source** has changed
Java	Java_Low_Visibility	Information_Exposure_Through_an_Error_Message	209	**Source** has changed
Java	Java_Low_Visibility	Information_Exposure_Through_Debug_Log	534	**Source** has changed
Java	Java_Low_Visibility	Information_Exposure_Through_Query_String	598	**Source** has changed
Java	Java_Low_Visibility	Information_Exposure_Through_Server_Log	533	**Source** has changed
Java	Java_Low_Visibility	Information_Leak_Through_Shell_Error_Message	535	**Source** has changed
Java	Java_Low_Visibility	Insufficiently_Protected_Credentials	522	**Source** has changed
Java	Java_Low_Visibility	Insufficient_Session_Expiration	613	**Source** has changed
Java	Java_Low_Visibility	Integer_Overflow	190	**Source** has changed
Java	Java_Low_Visibility	Integer_Underflow	191	**Source** has changed
Java	Java_Low_Visibility	JWT_Excessive_Expiration_Time	613	**Source** has changed
Java	Java_Low_Visibility	JWT_Use_Of_None_Algorithm	287	**Source** has changed
Java	Java_Low_Visibility	Logic_Time_Bomb	511	**Source** has changed
Java	Java_Low_Visibility	Object_Hijack	491	**Source** has changed
Java	Java_Low_Visibility	Portability_Flaw_Locale_Dependent_Comparison	474	**Source** has changed
Java	Java_Low_Visibility	Potential_ReDoS	400	**Source** has changed
Java	Java_Low_Visibility	Potential_ReDoS_In_Static_Field	400	**Source** has changed
Java	Java_Low_Visibility	Race_Condition	362	**Source** has changed
Java	Java_Low_Visibility	Reliance_on_Cookies_in_a_Decision	784	**Source** has changed
Java	Java_Low_Visibility	Reliance_on_DNS_Lookups_in_a_Decision	350	**Source** has changed
Java	Java_Low_Visibility	Sensitive_Cookie_in_HTTPS_Session_Without_Secure_Attribute	614	**Source** has changed
Java	Java_Low_Visibility	Serializable_Class_Containing_Sensitive_Data	499	**Source** has changed
Java	Java_Low_Visibility	TOCTOU	367	**Source** has changed
Java	Java_Low_Visibility	Unsynchronized_Access_To_Shared_Data	567	**Source** has changed
Java	Java_Low_Visibility	Use_of_Broken_or_Risky_Cryptographic_Algorithm	327	**Source** has changed
Java	Java_Low_Visibility	Use_Of_Hardcoded_Password	259	**Source** has changed
Java	Java_Low_Visibility	Use_Of_Hardcoded_Password_In_Config	260	**Source** has changed

Language	Group	Name	CWE	Changed Fields
Java	Java_Low_Visibility	Use_of_Hard_coded_Security_Constants	547	**Source** has changed
Java	Java_Low_Visibility	Use_of_RSA_Algorithm_without_OAEP	780	**Source** has changed
Java	Java_Low_Visibility	Using_Referer_Field_for_Authentication	293	**Source** has changed
Java	Java_Low_Visibility	UTF7_XSS	79	**Source** has changed
Java	Java_Medium_Threat	CGI_Stored_XSS	79	**Source** has changed
Java	Java_Medium_Threat	Cleartext_Submission_of_Sensitive_Information	319	**Source** has changed
Java	Java_Medium_Threat	DoS_by_Sleep	834	**Source** has changed
Java	Java_Medium_Threat	Download_of_Code_Without_Integrity_Check	494	**Source** has changed
Java	Java_Medium_Threat	Excessive_Data_Exposure	201	**Source** has changed
Java	Java_Medium_Threat	External_Control_of_Critical_State_Data	642	**Source** has changed
Java	Java_Medium_Threat	Frameable_Login_Page	829	**Source** has changed
Java	Java_Medium_Threat	Hardcoded_password_in_Connection_String	547	**Source** has changed
Java	Java_Medium_Threat	HttpOnlyCookies	1004	**Source** has changed
Java	Java_Medium_Threat	JSF_Managed_Bean_PII_Leak	359	**Source** has changed
Java	Java_Medium_Threat	JWT_Lack_Of_Expiration_Time	613	**Source** has changed
Java	Java_Medium_Threat	JWT_Sensitive_Information_Exposure	201	**Source** has changed
Java	Java_Medium_Threat	JWT_Use_Of_Hardcoded_Secret	798	**Source** has changed
Java	Java_Medium_Threat	Privacy_Violation	359	**Source** has changed
Java	Java_Medium_Threat	Process_Control	114	**Source** has changed
Java	Java_Medium_Threat	ReDoS_In_Pattern	400	**Source** has changed
Java	Java_Medium_Threat	Reliance_on_Cookies_without_Validation	565	**Source** has changed
Java	Java_Medium_Threat	Same_Seed_in_PRNG	336	**Source** has changed
Java	Java_Medium_Threat	SSL_Verification_Bypass	599	**Source** has changed
Java	Java_Medium_Threat	SSRF	918	**Source** has changed
Java	Java_Medium_Threat	Use_of_a_One_Way_Hash_without_a_Salt	759	**Source** has changed
Java	Java_Spring	Spring_CSRF	352	**Source** has changed
Java	Java_Spring	Spring_Missing_Content_Security_Policy	346	**Source** has changed
Java	Java_Spring	Spring_Missing_XSS_Protection_Header	693	**Source** has changed
Java	Java_Spring	Spring_Missing_X_Content_Type_Options	693	**Source** has changed
Java	Java_Spring	Spring_Missing_X_Frame_Options	1021	**Source** has changed
Java	Java_Spring	Spring_Use_of_Broken_or_Risky_Cryptographic_Primitive	327	**Source** has changed
Java	Java_Spring	Spring_Use_Of_Hardcoded_Password	259	**Source** has changed
JavaScript	JavaScript_APISecurity	NodeJS_Express_WebApi_GetApiList	0	**Source** has changed
Lua	Lua_High_Risk	Command_Injection	77	**Source** has changed
Lua	Lua_High_Risk	Reflected_XSS_All_Clients	79	**Source** has changed
Lua	Lua_High_Risk	Stored_Command_Injection	77	**Source** has changed
Lua	Lua_High_Risk	Stored_XSS	79	**Source** has changed
Lua	Lua_Low_Visibility	Command_Argument_Injection	78	**Source** has changed
Lua	Lua_Low_Visibility	Improper_Exception_Handling	248	**Source** has changed
Lua	Lua_Low_Visibility	Information_Exposure_Through_Server_Log	359	**Source** has changed
Lua	Lua_Low_Visibility	Insufficient_Session_Expiration	613	**Source** has changed
Lua	Lua_Low_Visibility	JWT_No_Expiration_Time_Validation	613	**Source** has changed
Lua	Lua_Low_Visibility	JWT_No_NotBefore_Validation	304	**Source** has changed
Lua	Lua_Low_Visibility	Missing_Framing_Policy	1021	**Source** has changed
Lua	Lua_Low_Visibility	Null_Pointer_Dereference	457	**Source** has changed
Lua	Lua_Low_Visibility	Password_In_Comment	615	**Source** has changed
Lua	Lua_Low_Visibility	Reliance_on_DNS_Lookups_in_a_Decision	350	**Source** has changed

Language	Group	Name	CWE	Changed Fields
Lua	Lua_Low_Visibility	Stored_Command_Argument_Injection	78	**Source** has changed
Lua	Lua_Low_Visibility	Using_Referer_Field_for_Authentication	287	**Source** has changed
Lua	Lua_Medium_Threat	DoS_by_Sleep	834	**Source** has changed
Lua	Lua_Medium_Threat	DoS_from_Evil_Regex	400	**Source** has changed
Lua	Lua_Medium_Threat	DoS_from_RegEx_Injection	400	**Source** has changed
Lua	Lua_Medium_Threat	JWT_Use_Of_Hardcoded_Secret	798	**Source** has changed
Lua	Lua_Medium_Threat	Misconfigured_HSTS_Header	346	**Source** has changed
Lua	Lua_Medium_Threat	Open_Redirect	601	**Source** has changed
Lua	Lua_Medium_Threat	Privacy_Violation	359	**Source** has changed
PHP	PHP_High_Risk	Deserialization_of_Untrusted_Data	502	**Source** has changed
PHP	PHP_High_Risk	Stored_XPath_Injection	643	**Source** has changed
Python	Python_Medium_Threat	Object_Access_Violation	610	**Source** has changed
Scala	Scala_Medium_Threat	Use_of_a_One_Way_Hash_without_a_Salt	759	CxDescriptionId changed from 345 to **2576**, **Source** has changed
Scala	Scala_Medium_Threat	Use_of_a_One_Way_Hash_with_a_Predictable_Salt	760	**Source** has changed
Dart	Dart_Mobile_Medium_Threat	Relative_Path_Traversal	23	Name changed from Path_Traversal to **Relative_Path_Traversal**, CWE changed from 22 to **23**, **Source** has changed
Lua	Lua_Low_Visibility	Privacy_Violation_in_JWT	200	PackageId changed from 1518 to **1517**, Name changed from JWT_Sensitive_Information_Exposure to **Privacy_Violation_in_JWT**, CWE changed from 359 to **200**, CxDescriptionId changed from 3094 to **4199**, Severity changed from 2 to **1**, **Source** has changed, Group changed from Lua_Medium_Threat to **Lua_Low_Visibility**
Objc	Apple_Secure_Coding_Guide	Jailbreak_File_Referenced_By_Name	668	Name changed from Jailbrake_File_Referenced_By_Name to **Jailbreak_File_Referenced_By_Name**
Python	Python_Medium_Threat	ReDoS_Injection	400	PackageId changed from 1378 to **1379**, Severity changed from 1 to **2**, **Source** has changed, Group changed from Python_Low_Visibility to **Python_Medium_Threat**

Changed Source:

CPP / CPP_Best_Coding_Practice / Buffer_Size_Literal_Condition

Code changes

```
---
+++
@@ -10,6 +10,11 @@
 CxList ArrayDefinition = Find_ArrayCreateExpr();
 CxList unkRefs = Find_Unknown_References();
 CxList foundSizeInt = All.NewCxList();
+CxList customAttribute = Find_CustomAttribute().FindByShortName("CxNotLiteralBufferSize").GetFathers();
+customAttribute.Add(customAttribute.FindByType<CustomAttributeCollection>().GetFathers());
+
+ArrayDefinition -= ArrayDefinition.GetByAncs(customAttribute);
+
 //find arrays that are initialized with an int as a size
 foreach(CxList arrayDef in ArrayDefinition)
 {
```

CPP / CPP_Buffer_Overflow / Buffer_Improper_Index_Access

Code changes

```
---
+++
@@ -1,5 +1,6 @@
 // This query complements the results of the Off_By_One query.
 CxList unkRefs = Find_Unknown_References();
+CxList ints = Find_Integer_Literals();
```

```
 CxList inputs = Find_Inputs();
 CxList methods = Find_Methods();
 CxList conditions = Find_Conditions();
@@ -8,6 +9,8 @@
 CxList allDeclarators = Find_Declarators();
 allDeclarators.Add(paramDeclarators);
 CxList binExpr = Find_BinaryExpr();
+CxList ifStmts = Find_Ifs();
+CxList strlenCalls = methods.FindByShortNames("strlen");

 // 1. General Improper Index Access Usage
 CxList nodes = Find_Improper_Index_Access(false);
@@ -18,6 +21,30 @@
 CxList inputNodeIndexerRef = inputNodesUnkRefs.GetAncOfType<IndexerRef>();
 nodes -= inputNodeIndexerRef;
 nodes.Add(inputsNodes);
+
+// Sanitize commonly used structs from libraries
+nodes -= indexAccess.FindByShortNames("s6_addr");
+
+// Remove nodes where length is checked in if statement
+CxList sizeComp = ints.GetAncOfType<BinaryExpr>() * strlenCalls.GetAncOfType<BinaryExpr>();
+CxList sizeCompInIf  = sizeComp.GetByAncs(ifStmts);
+
+foreach(CxList sizeComparison in sizeCompInIf)
+{
+   CxList sizeIf = sizeComparison.GetAncOfType<IfStmt>();
+
+   CxList maxSize = ints.GetByAncs(sizeComparison);
+   IntegerLiteral maxSizeInt = maxSize.TryGetCSharpGraph<IntegerLiteral>() as IntegerLiteral;
+
+   CxList strlenNodes = unkRefs.GetParameters(strlenCalls.GetByAncs(sizeComparison));
+   CxList checkedNodes = nodes * indexAccess.GetByAncs(sizeIf).FindAllReferences(strlenNodes);
+
+   CxList checkedIndex = checkedNodes.CxSelectElements<IndexerRef>(_ => _.Indices, 0);
+   IntegerIntervalAbstractValue size = new IntegerIntervalAbstractValue(0, maxSizeInt.Value - 1);
+   CxList validNodes = checkedIndex.FindByAbstractValue(_ => _.IncludedIn(size)).GetFathers();
+   nodes -= validNodes;
+}
+
 result = nodes;

 /* 2. Accessing input string by length without sanitizing
@@ -30,7 +57,7 @@

 // 2.1 Inputs
 // Get all unknown references that are used as parameters of strlen
-CxList strlenCalls = methods.FindByShortNames("strlen");
```

```
+
 CxList strings = unkRefs.FindAllReferences(unkRefs.GetParameters(strlenCalls, 0));
 // Get those that are paramDecl and inputs
 CxList stringParameters = strings.GetAncOfType<ParamDecl>() * inputs;
@@ -58,13 +85,13 @@
 result.Add(stringParameters.InfluencingOnAndNotSanitized(
    indexRefs.CxSelectDomProperty<IndexerRef>(x => x.Target),
    sanitizers));
-
 /* 3. Accessing array by input value without sanitizing */

 // unkRefs used as indices in array accesses
 CxList indices = indexAccess.CxSelectElements<IndexerRef>(x => x.Indices).FindByType<UnknownReference>();

 CxList idxSanitizers = All.NewCxList();
+
 // Sanitize address passed variables
 CxList addressParams = unkRefs.GetByAncs(Find_Unarys().FilterByDomProperty<UnaryExpr>(u => u.Operator == UnaryOperator.Address));

@@ -82,17 +109,16 @@
    assignSanitizers.CxSelectDomProperty<AssignExpr>(a => a.Left),
    sanitizerBinOps.GetAncOfType<Declarator>());
 // Create the sanitization list
+// strlen is considered a sanitizer, as well as methods starting with test (heuristic)
 idxSanitizers.Add(
    assignSanitizersUnkRefs,
    addressParams,
    unkRefs.GetByAncs(sanitizerBinOps),
-   methods.FindByShortNames("strlen", "test*")  // strlen is considered a sanitizer, as well as methods starting with test (heuristic)
+   methods.FindByShortNames("strlen", "test*")
    );

 // get only indices influenced by inputs and not sanitized
 indices = indices.InfluencedByAndNotSanitized(inputs, idxSanitizers).GetLastNodesInPath();
-// all references to those unkRefs
-CxList indicesRefs = unkRefs.FindAllReferences(indices);
 // paths from inputs to conditions
 CxList inputToCondition = inputs.DataInfluencingOn(conditions);
 // those referencess that are used in a condition
```

CPP / CPP_Buffer_Overflow / Buffer_Overflow_LongString

Code changes

```
---
+++
@@ -35,6 +35,7 @@
 CxList flow = relevant.InfluencedByAndNotSanitized(stringLiteral, sanitizers);

 CxList arrayCreateExpr = Find_ArrayCreateExpr();
```

```
+CxList customAttribute = Find_CustomAttribute().FindByShortName("CxNotLiteralBufferSize").GetAncOfType<VariableDeclStmt>();

 CxList valueAccess = varsOfTypeCharPointer.FindByFathers(varsOfTypeCharPointer.GetFathers().FindByShortName("Pointer"));

@@ -55,7 +56,8 @@
            {
                continue;
            }
-           if(r.Sizes.Count > 0)
+           int isLiteralSize = arr.GetByAncs(customAttribute).Count;
+           if(isLiteralSize != 1)
            {
                ExpressionCollection sizes = r.Sizes;
                CxList firstSize = All.NewCxList();
```

CPP / CPP_Buffer_Overflow / Buffer_Overflow_Wrong_Buffer_Size

Code changes

```
---
+++
@@ -4,11 +4,11 @@
 CxList integers = Find_Integers();
 integers.Add(Find_Integer_Literals());

-CxList nodesWithAbsVal = All.NewCxList();
-nodesWithAbsVal.Add(unknownRefs);
-nodesWithAbsVal.Add(Find_Strings());
-nodesWithAbsVal.Add(Find_CharLiteral());
-nodesWithAbsVal.Add(integers);
+CxList customAttribute = Find_CustomAttribute().FindByShortName("CxNotLiteralBufferSize").GetFathers();
+customAttribute.Add(customAttribute.FindByType<CustomAttributeCollection>().GetFathers());
+CxList arrays = Find_ArrayCreateExpr().GetByAncs(customAttribute);
+
+CxList nodesWithAbsVal = All.NewCxList(unknownRefs, Find_Strings(), Find_CharLiteral(), integers);

 // Helper delegate to compare two parameters as for their AbsValue
 Func <CxList, string, CxList, bool> CompareExprsByAbsVal = delegate(CxList fstParam, string compareOp, CxList sndParam){
@@ -63,9 +63,7 @@
 };

 // Inputs
-CxList inputs = Find_Unbounded_Inputs();
-inputs.Add(Find_Read());
-inputs.Add(Find_DB());
+CxList inputs = All.NewCxList(Find_Unbounded_Inputs(), Find_Read(), Find_DB());

 // Sanitizers
 CxList sizeofMethods = methodInvokes.FindByShortName("sizeof");
@@ -98,13 +96,9 @@
```

```
 CxList vulnerableMethods = All.FindByShortNames(vulnerableMethodsNames);

-CxList sizeParameters = All.NewCxList();
-sizeParameters.Add(sizeofParams);
-sizeParameters.Add(strlenParams);
+CxList sizeParameters = All.NewCxList(sizeofParams, strlenParams);

-CxList sizeWithValue = All.NewCxList();
-sizeWithValue.Add(nodesWithAbsVal);
-sizeWithValue.Add(integers);
+CxList sizeWithValue = All.NewCxList(nodesWithAbsVal, integers);

 foreach(CxList method in vulnerableMethods)
 {
@@ -147,6 +141,9 @@
        // Keep useful nodes
        CxList sizeWithSize = sizeWithValue.GetByAncs(size);
        CxList destinationWithSize = nodesWithAbsVal.GetByAncs(destination);
+
+       if(arrays.InfluencingOn(destinationWithSize).Count > 0)
+           continue;

        // Check if the size parameter contains size/strlen methods, which act as sanitizers
        CxList sizeInfluencedBySanitizer = sizeWithSize.InfluencedByAndNotSanitized(sizeParameters, Find_ParamDecl());
```

CPP / CPP_Buffer_Overflow / Improper_Null_Termination

Code changes

```
---
+++
@@ -4,11 +4,6 @@
 CxList unkRefs = Find_Unknown_References();
 CxList declarators = Find_Declarators();
 CxList allArrayInitializer = Find_ArrayInitializer();
-
-CxList allRelevantTypes = All.NewCxList(
-   Find_Builtin_Char_Types(),
-   Find_Builtin_Char_Microsoft_Types()
-   );

 CxList declsParams = All.NewCxList(
    declarators,
@@ -17,15 +12,31 @@

 // Common functions that do not copy the null terminator.
 List<string> methodWithoutNullTermStr = new List<string>{
-       "strncpy","wcsncpy","mbstowcs", "mbsrtowcs","wcstombs",
-       "wcsrtombs", "_strncpy_l","_tcsncpy","_tcsncpy_l","_tcsncat","_tcsncat_l"
```

```
+       "strncpy",
+       "wcsncpy",
+       "mbstowcs",
+       "mbsrtowcs",
+       "wcstombs",
+       "wcsrtombs",
+       "_strncpy_l",
+       "_tcsncpy",
+       "_tcsncpy_l",
+       "_tcsncat",
+       "_tcsncat_l"
        };

 CxList methodWithoutNullTerm = methods.FindByShortNames(methodWithoutNullTermStr);
 CxList declWithNullTerm = allArrayInitializer.GetAncOfType<Declarator>();

 // Add other types of parameters references
-CxList supportedParamTypes = declsParams.FindByTypes(new string[] {"char", "png_bytep", "OLECHAR", "wchar_t", "TCHAR", "void"});
+CxList supportedParamTypes = declsParams.FindByTypes(new string[] {
+   "char",
+   "png_bytep",
+   "OLECHAR",
+   "wchar_t",
+   "TCHAR",
+   "void"});
+
 CxList supportedParamTypesRefs = All.NewCxList(
    unkRefs.FindAllReferences(supportedParamTypes),
    Find_CastExpr(),
@@ -36,16 +47,7 @@

 CxList charToCheckReqNullTerm = supportedParamTypesRefs.GetParameters(methodWithoutNullTerm, 0);

-CxList declUnk = All.NewCxList(unkRefs, declarators);
-allRelevantTypes.Add(declUnk.FindByPointerTypes(new string[] {"char*", "BYTE*", "ValueType.Ch" }));
-
-CxList relevantParams = unkRefs.FindAllReferences(allRelevantTypes);
-
-CxList methodsToCheck = methods.FindByShortNames("fread");
-CxList memCpyParams = relevantParams.GetParameters(methodsToCheck, 0);
-
 CxList funcsNames = All.NewCxList(
-   methodsToCheck.GetAncOfType<MethodDecl>(),
    methodWithoutNullTerm.GetAncOfType<MethodDecl>(),
    methodWithoutNullTerm.GetAncOfType<ConstructorDecl>()
    );
@@ -57,7 +59,7 @@
    var[4] = false;
    var[4] = NULL;

 */

-CxList nullTermination = All.NewCxList();

+

 IAbstractValue zero = new IntegerIntervalAbstractValue(0);

 CxList assignZero = All.FindByAbstractValue(

    abstractValue => zero.IncludedIn(abstractValue) && abstractValue is not AnyAbstractValue

@@ -65,7 +67,7 @@

 CxList assignFalse = Find_BooleanLiteral().FindByAbstractValue(abstractValue => abstractValue is FalseAbstractValue);

 CxList assignNull = Find_NullLiteral().FindByAbstractValue(abstractValue => abstractValue is NullAbstractValue);


-nullTermination.Add(

+CxList nullTermination = All.NewCxList(

    assignZero,

    assignNull,

    assignFalse,

@@ -79,8 +81,7 @@

    }

 }


-CxList allRelevantParams = All.NewCxList();

-allRelevantParams.Add(memCpyParams, charToCheckReqNullTerm);

+CxList allRelevantParams = All.NewCxList(charToCheckReqNullTerm);

 CxList allRefsOfRelevantParams = unkRefs.FindAllReferences(allRelevantParams);

 CxList sanitizers = allRefsOfRelevantParams.GetByAncs(nullTermination.GetAncOfType<AssignExpr>());


@@ -101,7 +102,8 @@

 sanitizerRefs.Add(methods.FindByShortName("calloc").GetAssignee());

 sanitizerRefs = allRelevantParams.InfluencedBy(sanitizerRefs).GetLastNodesInPath();

 sanitizers.Add(allRelevantParams.FindAllReferences(sanitizerRefs));

-//All the char parameters that are declared as unsined should be removed (they are, probably, declared as unsigned to be used as numbers not chars of string)

+// All the char parameters that are declared as unsined should be removed

+// (they are, probably, declared as unsigned to be used as numbers not chars of string)

 sanitizers.Add(allRelevantParams.FindByType("char").FindByTypeModifiers(TypeSignednessModifiers.Unsigned));


 // strncpy sanitizer => strnctyp(ret, sanitized), sprintf(sanitized, "value")

@@ -112,5 +114,4 @@


 // Flow and outputs

 result = charToCheckReqNullTerm;

-result.Add(memCpyParams);

 result -= sanitizers;
```

CPP / CPP_Medium_Threat / Memory_Leak

Code changes

```
---

+++

@@ -5,21 +5,38 @@
```

```
 throwStmts.Add(Find_ThrowExpr());
 CxList destructorDecls = Find_DestructorDecl();
 CxList decls = Find_VariableDeclStmt();
-decls.Add(Find_FieldDecls());
+CxList fieldDecls = Find_FieldDecls();
+decls.Add(fieldDecls);
 CxList declarators = Find_Declarators();
 CxList methods = Find_Methods();
 CxList parms = Find_Parameters().CxSelectDomProperty<Param>(x => x.Value);
+CxList fieldDeclaration = declarators.GetByAncs(fieldDecls);
 CxList references = All.NewCxList();
 references.Add(unknownRefs, Find_IndexerRefs());

 // 0. Arrays that use "delete" (deletes single object) and not "delete[]" (deletes group of objects)
 CxList arrayCreateExpr = memoryAllocation.FindByType<ArrayCreateExpr>();
 // These are single object deletes
-CxList singleDeletes = Find_String_Literal().GetParameters(methods).FindByName("singleObjDel").GetAncOfType<MethodInvokeExpr>();
+CxList singleDeletes = Find_String_Literal().GetParameters(methods).FindByName("singleObjDel").
+   GetAncOfType<MethodInvokeExpr>();
 // These are array deletes
 CxList arrayDeletes = memoryDeallocation.FindByShortName("delete").FindByNumberOfParameters(1);
 // Remove proper deallocated arrays
 arrayCreateExpr -= arrayDeletes.DataInfluencedBy(arrayCreateExpr).GetFirstNodesInPath();
+
+// 1. Find deletes inside destructors for news where there is no flow
+// Find assignments involving array creation expressions
+CxList assign = arrayCreateExpr.GetAncOfType<AssignExpr>();
+CxList leftSide = assign.CxSelectDomProperty<AssignExpr>(a => a.Left);
+// Find array deletes influenced by destructor declarations
+CxList deletesAncs = arrayDeletes.GetByAncs(destructorDecls);
+// Find variables to be deleted
+CxList deleteVars = unknownRefs.GetParameters(deletesAncs);
+CxList declsDelete = fieldDeclaration.FindDefinition(deleteVars);
+
+// Find variables that are assigned to array creation and have reference to
+// a destructor method that has a delete method
+CxList varToDelete = leftSide.FindAllReferences(declsDelete);

 CxList delDeallocation = All.NewCxList();
 foreach(CxList arrayExpr in arrayCreateExpr){
@@ -28,8 +45,7 @@
        delDeallocation.Add(arrayExpr.GetAssignee());
    }
 }
-
-// 1. Throw statement and expressions between memory allocations and memory deallocation
+// 1.1 Throw statement and expressions between memory allocations and memory deallocation
 CxList throwAfterAllocation = All.NewCxList();
 foreach(CxList allocation in memoryAllocation)
 {
@@ -61,7 +77,8 @@
 // 4. Free inside class destructor
 // 4.1 Exact matches from allocation/deallocation
 CxList allAllocatedReferences = references.FindAllReferences(allocatedReferences);
-CxList deallocatedInDestructor = allAllocatedReferences.GetByAncs(parms.GetParameters(memoryDeallocation.GetByAncs(destructorDecls)));
+CxList deallocatedInDestructor = allAllocatedReferences.GetByAncs(parms.GetParameters(memoryDeallocation.
+   GetByAncs(destructorDecls)));
 allocatedReferences -= allocatedReferences.FindAllReferences(deallocatedInDestructor);

 // 4.2 No flow accessible references
@@ -92,5 +109,8 @@
 // when the object is no longer needed)
 allocatedReferences -= allocatedReferences.FindByType("auto_ptr");

+// Remove variables that have destructor assigned to array creation expressions to be deleted
+delDeallocation -= varToDelete;
+
 result = allocatedReferences;
 result.Add(delDeallocation, memoryAllocation, throwAfterAllocation);
```

CPP / CPP__MISRA__C__2012 / R03__X__Comments

Code changes

```
---
+++
@@ -21,7 +21,7 @@

 result.Add(
    // Single line comments ending with line splice
-   comments.FilterByDomProperty<Comment>(_ => _.ShortName.StartsWith("//") && Regex.Matches(_.ShortName, @"\\\s+$").Count > 0),
+   comments.FilterByDomProperty<Comment>(_ => _.ShortName.StartsWith("//") && Regex.Matches(_.ShortName, @"\\\s*$").Count > 0),
    // slash star inside single line
    comments.FilterByDomProperty<Comment>(_ => Regex.Matches(_.ShortName, @"^//.+/\*").Count > 0),
    // multiline with slash slash or slash star
```

CSharp / CSharp__APISecurity / CSharp__WebApi__GetApiList

Code changes

```
---
+++
@@ -53,13 +53,15 @@
 methodDecls = methodDecls.FindByFieldAttributes(Modifiers.Public);
 List <string> annotations = new List<string>() {
        "HttpGet", "HttpPost", "HttpPut", "HttpDelete", "HttpPatch", "HttpHead", "HttpOptions"};
+List <string> contentResponseTypeAnnotations = new List <string>() {"Produces", "Consumes"};
 CxList allMapRoute = mthds.FindByShortNames(new List<string>{"MapRoute","MapHttpRoute", "MapControllerRoute",
        "MapDefaultControllerRoute"});
 CxList specialMapRoute = mthds.FindByShortNames(new List<string>{"MapGet","MapPost", "MapPut", "MapDelete"});
 // Find HttpGet, HttpPost, HttpPut and HttpDelete annotations

 CxList allRequestAnnotations = customAttributes.FindByShortNames(annotations);

+CxList allResponseContentTypeAnnotations = customAttributes.FindByShortNames(contentResponseTypeAnnotations);

 CxList allAnnotations = All.NewCxList();

-allAnnotations.Add(allRequestAnnotations);

+allAnnotations.Add(allRequestAnnotations, allResponseContentTypeAnnotations);

 //Find Route annotations

 CxList allRouteAnnotations = customAttributes.FindByShortName("Route");

 allAnnotations.Add(allRouteAnnotations);
```

CSharp / CSharp_High_Risk / Deserialization_of_Untrusted_Data_MSMQ

Code changes

CSharp / CSharp_High_Risk / Reflected_XSS_All_Clients

Code changes

```
---

+++

@@ -1,4 +1,4 @@

-if(All.isWebApplication || Check_Web_Application().Any())

+if((All.isWebApplication || Check_Web_Application().Any()) && Find_ASP_XSS_Safe().Count == 0)

 {

    CxList methods = Find_Methods();

    CxList parameters = Find_Parameters();
```

CSharp / CSharp_High_Risk / Stored_XSS

Code changes

```
---

+++

@@ -1,4 +1,4 @@

-if(All.isWebApplication || Check_Web_Application().Count() > 0)

+if((All.isWebApplication || Check_Web_Application().Count() > 0) && Find_ASP_XSS_Safe().Count == 0)

 {

    CxList inputs = Find_Stored_Inputs();

    CxList outputs = Find_XSS_Outputs();
```

CSharp / CSharp_Medium_Threat / Unsafe_Object_Binding

Code changes

```
---

+++

@@ -12,7 +12,6 @@

    /*

        GET INPUTS

    */

-

    CxList inputs = All.NewCxList();


    //Parameters related to public methods
```

```
@@ -40,7 +39,6 @@
    /*
        SaveChanges
    */
-
    CxList saveChanges = methodsControllers.FindByShortNames(new List<string> {"SaveChanges", "SaveChangesAsync"}, true);
    CxList dbSaveChanges = saveChanges * methodsControllers;

@@ -54,7 +52,6 @@

    //Remove update methods with array creations as parameters
    foreach(CxList updateMethod in updateMethods){
-
        CxList updateParameters = parameters.GetParameters(updateMethod);
        CxList parametersArray = updateParameters * arrayCreationParams;

@@ -72,7 +69,6 @@
        CxList following = update.GetAncOfType<ExprStmt>().GetFollowingStatements();

        while(following.Count > 0){
-
            CxList saveInvoc = dbSaveChanges.GetByAncs(following);

            if(saveInvoc.Count > 0){
@@ -91,12 +87,13 @@
    CxList safeParams = unkRefs.FindAllReferences(unkRefs.GetParameters(safeUpdateMethods, 0));
    CxList safeDbInsertion = safeParams.InfluencingOn(methodsControllers.FindByShortName("Entry")).GetLastNodesInPath();
    inputsNotUsed -= inputsNotUsed.InfluencingOn(safeDbInsertion);
+
    foreach(CxList input in inputsNotUsed){
        CxList saveChange = dbSaveChanges.GetByAncs(input.GetAncOfType<MethodDecl>());
-       if(saveChange.Count > 0)
+       CxList inputInfluencingSave = input.InfluencingOn(saveChange.GetTargetOfMembers());
+       if(saveChange.Count > 0 && inputInfluencingSave.Count > 0)
            result.Add(input.ConcatenatePath(saveChange, false));
    }
-
 }

 result.Add(Find_Unsafe_Object_Binding_NetCore());
```

Dart / Dart_Mobile_Best_Coding_Practice / WebView_Cache_Information_Leak

Code changes

```
---
+++
@@ -1,15 +1,11 @@
 if(cxXPath.GetXmlFiles("AndroidManifest.xml").Count() > 0) {
-   CxList classes = Find_ClassDecl();
-   CxList methodDecls = Find_MethodDecls();
-   CxList methodInvks = Find_Methods();
+   CxList webViews = Find_WebView_Javascript_Injection_Sinks();
+   CxList unknownReferences = Find_UnknownReference();

-   CxList webViews = Find_WebView_Javascript_Injection_Sinks();
+   CxList controllersRefs = unknownReferences.FindAllReferences(webViews.GetTargetOfMembers());
+   CxList sanitizedControllers = controllersRefs.GetMembersOfTarget().FindByShortName("clearCache").GetTargetOfMembers();
+   CxList sanitizedControllersRefs = controllersRefs.FindAllReferences(sanitizedControllers);
+   CxList unsanitizedWebViews = (webViews.GetTargetOfMembers() - sanitizedControllersRefs).GetMembersOfTarget();

-   CxList webViewsClass = classes.GetClass(webViews);
-   CxList webViewClassLifecycle = methodDecls.FindByShortName("didChangeAppLifecycleState");
-   CxList clearCache = methodInvks.FindByShortName("clearCache");
-   CxList sanitizers = clearCache.GetByAncs(webViewClassLifecycle).GetAncOfType<ClassDecl>() * webViewsClass;
-   CxList vulnerable = webViewsClass - sanitizers;
-
-   result = webViews.GetByAncs(vulnerable);
+   result = unsanitizedWebViews;
 }
```

Dart / Dart_Mobile_High_Risk / Unencrypted_Sensitive_Information_in_Publicly_Accessible_Cloud_Storage

Code changes

```
---
+++
@@ -1,8 +1,11 @@
-CxList inputs = Find_Sensitive_Information();
-
-CxList sanitizers = Find_Encryption();
-sanitizers.Add(Find_Hashing());
-
-CxList outputs = Find_Firebase_Cloud_Sinks();
-
-result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceSmallFlow);
+if (Find_Firebase_Auth().Count == 0) //Firebase authentication is the sanitizer for cloud storage. Public accessibility becomes private.
+{
+   CxList inputs = Find_Sensitive_Information();
+
+   CxList sanitizers = Find_Encryption();
+   sanitizers.Add(Find_Hashing());
+
+   CxList outputs = Find_Firebase_Cloud_Sinks();
+
+   result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceSmallFlow);
+}
```

Dart / Dart_Mobile_High_Risk / Unsafe_Reflection

Code changes

```diff
---
+++
@@ -3,24 +3,17 @@
 CxList unkRefs = Find_UnknownReference();
 CxList imports = Find_Import();

-CxList possibleElems = All.NewCxList();
-possibleElems.Add(unkRefs, methods, Find_ClassDecl(), Find_ConstantDecl(), Find_MemberAccesses());
-CxList unkDecl = All.NewCxList();
-unkDecl.Add(unkRefs, Find_Declarators());
-
-bool reflected_flutter = false;
-/*
-   Inputs - Dart
-*/
-CxList inputs = Find_UI_Inputs();
-inputs.Add(Find_Public_Storage_Inputs(), Find_Private_Storage_Inputs(), Find_Cloud_Firestore_Inputs());
-
-// Remote Inputs
-inputs.Add(Find_Remote_Inputs());
+CxList possibleElems = All.NewCxList(unkRefs, methods, Find_ClassDecl(), Find_ConstantDecl(), Find_MemberAccesses());
+CxList unkDecl = All.NewCxList(unkRefs, Find_Declarators());

 /*
-   Inputs - Flutter
+   Inputs
 */
+CxList inputs = All.NewCxList(Find_UI_Inputs(), Find_Remote_Inputs(),
+   Find_Public_Storage_Inputs(), Find_Private_Storage_Inputs());
+
+// Check if Reflectable package is imported (more sinks)
+bool reflected_flutter = false;
 CxList importReflectable = imports
    .FilterByDomProperty<Import>(imp => imp.ImportedFilename.EndsWith("reflectable.dart"));
 CxList importNamespace = importReflectable.GetAncOfType<NamespaceDecl>();
@@ -28,73 +21,70 @@
 CxList reflectable = Find_BaseRef().FindByShortName("Reflectable");
 CxList reflectableNamespace = reflectable.GetAncOfType<NamespaceDecl>();

-CxList reflector = unkRefs.NewCxList();
+CxList reflector = All.NewCxList();
 CxList reflectorRefs = All.NewCxList();
-if((importNamespace * reflectableNamespace) != null){
+if ((importNamespace * reflectableNamespace) != null)
+{
    reflected_flutter = true;
    reflector = reflectable.GetAncOfType<ClassDecl>();
    reflectorRefs = possibleElems.FindByType(reflector);
-   inputs.Add(reflectorRefs);
 }

 /*
    Sinks
 */
-CxList sinks = All.NewCxList();
-
-//libMirror.declarations
+// LibraryMirror.declarations
 CxList libMirror = unkDecl.FindByType("LibraryMirror");
-CxList libMirrorMethods = libMirror.GetMembersOfTarget().FindByShortNames(new string[]{"declarations"});
+// Also add LibraryMirrors from Reflectable package
+if (reflected_flutter)
+   libMirror.Add(Find_All_Instances(reflectorRefs.GetMembersOfTarget().FindByShortName("findLibrary")));
+CxList libMirrorMethods = libMirror.GetMembersOfTarget().FindByShortName("declarations");
 CxList libMirrorIndexerRef = libMirrorMethods.GetAncOfType<IndexerRef>();
 CxList libMirrorIndexes = libMirrorIndexerRef.CxSelectElements<IndexerRef>(x => x.Indices, 0);

-//get/set Mirror
+// Mirror get/set
 CxList classElem = unkDecl.FindByType("ClassMirror");
 CxList instanceElem = unkDecl.FindByType("InstanceMirror");
-CxList classAndInstance = All.NewCxList();
-classAndInstance.Add(instanceElem, classElem);
+CxList classAndInstance = All.NewCxList(instanceElem, classElem);

 CxList classInstanceTarget = classAndInstance.GetMembersOfTarget();

 CxList getters = classInstanceTarget.FindByShortName("getField");
 CxList getParams = parameters.GetParameters(getters);
-CxList getParamChilds = All.FindByFathers(getParams);
+CxList getParamChilds = All.FindByFathers(getParams);

 CxList setters = classInstanceTarget.FindByShortName("setField");
 CxList setParams = parameters.GetParameters(setters, 0);
-CxList setParamChilds = All.FindByFathers(setParams);
+CxList setParamChilds = All.FindByFathers(setParams);

-//invoke
+// invoke
 List <string> invokes = new List<string>(){ "invoke" };
-if(reflected_flutter){
+if (reflected_flutter)
    invokes.Add("invoker");
-}

 CxList invokeMethods = methods.FindByShortNames(invokes);
 CxList invokeGetTarget = invokeMethods.GetTargetOfMembers();

 CxList invokeTargets = classAndInstance.DataInfluencingOn(invokeGetTarget).GetLastNodesInPath();


 CxList libInvoke = unkRefs.FindAllReferences(libMirror).GetMembersOfTarget() * invokeMethods;

-CxList invokeSinks = All.NewCxList();

-invokeSinks.Add(invokeTargets.GetMembersOfTarget(), libInvoke);

+CxList invokeSinks = All.NewCxList(invokeTargets.GetMembersOfTarget(), libInvoke);


-

-sinks.Add(getParamChilds, setParamChilds, invokeSinks, libMirrorIndexes);

+CxList sinks = All.NewCxList(getParamChilds, setParamChilds, invokeSinks, libMirrorIndexes);


 /*

    Sinks - Flutter

 */

-

-if(reflected_flutter){

-   //invokeGetter

+if (reflected_flutter)

+{

+   // invokeGetter

    CxList invokeGetters = classInstanceTarget.FindByShortName("invokeGetter");

    CxList getInvokeParams = parameters.GetParameters(invokeGetters);

-   CxList getInvokeParamChilds = All.FindByFathers(getInvokeParams);

+   CxList getInvokeParamChilds = All.FindByFathers(getInvokeParams);


-   //invokeSetter

+   // invokeSetter

    CxList invokeSetters = instanceElem.GetMembersOfTarget().FindByShortName("invokeSetter");

    CxList setInvokeParams = parameters.GetParameters(invokeSetters, 0);

-   CxList setInvokeParamChilds = All.FindByFathers(setInvokeParams);

+   CxList setInvokeParamChilds = All.FindByFathers(setInvokeParams);


    sinks.Add(getInvokeParamChilds, setInvokeParamChilds);

 }

@@ -103,8 +93,8 @@

    Sanitizers

 */

 CxList whiteListConditions = Find_WhiteListSanitizers();

+CxList inputsSanitized = inputs.DataInfluencingOn(whiteListConditions).GetFirstNodesInPath();

+inputs -= inputsSanitized;


-CxList inputsSanitized = inputs.DataInfluencingOn(whiteListConditions).GetFirstNodesInPath();

-

-inputs = inputs - inputsSanitized;

 result = sinks.InfluencedBy(inputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

+result.Add(inputs * sinks);
```

Dart / Dart__Mobile_Low_Visibility / Encrypted__Sensitive__Information__in__Publicly__Accessible__Cloud__Storage

Code changes

```
---
+++
@@ -1,10 +1,12 @@
-CxList inputs = Find_Sensitive_Information();
+if (Find_Firebase_Auth().Count == 0) //Firebase authentication is the sanitizer for cloud storage. Public accessibility becomes private.
+{
+   CxList inputs = Find_Sensitive_Information();

-CxList encryptHash = Find_Encryption();
-encryptHash.Add(Find_Hashing());
+   CxList encryptHash = Find_Encryption();
+   encryptHash.Add(Find_Hashing());

-CxList encryptHashInputs = encryptHash.InfluencedBy(inputs).GetFirstNodesInPath();
-
-CxList outputs = Find_Firebase_Cloud_Sinks();
-
-result = outputs.InfluencedBy(encryptHashInputs).ReduceFlow(CxList.ReduceFlowType.ReduceSmallFlow);
+   CxList encryptHashInputs = encryptHash.InfluencedBy(inputs).GetFirstNodesInPath();
+   CxList outputs = Find_Firebase_Cloud_Sinks();
+
+   result = outputs.InfluencedBy(encryptHashInputs).ReduceFlow(CxList.ReduceFlowType.ReduceSmallFlow);
+}
```

Dart / Dart__Mobile_Low_Visibility / Unencrypted__Sensitive__Information__in__Internal__Storage

Code changes

```
---
+++
@@ -3,6 +3,6 @@
 CxList sanitizers = Find_Encryption();
 sanitizers.Add(Find_Hashing());

-CxList outputs = Find_Private_File_Storage();
+CxList outputs = Find_Private_Storage_Sinks();

 result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceSmallFlow);
```

Dart / Dart__Mobile_Low_Visibility / Unencrypted__Sensitive__Information__in__Temporary__File

Code changes

```
---
+++
@@ -1,7 +1,13 @@
 CxList methdInvk = Find_Methods();

 //inputs
-CxList inputs = methdInvk.FindByShortNames("createTemp", "createTempSync", "getTemporaryDirectory", "systemTemp");
```

```
+// "Directory.systemTemp.createTemp()/createTempSync()" and "getTemporaryDirectory()"
+CxList inputs = methdInvk.FindByShortNames("createTemp", "createTempSync", "getTemporaryDirectory");
+// "Directory.systemTemp" in itself is also an input
+CxList directorySystemTemps = Find_UnknownReference().FindByType("Directory").GetMembersOfTarget()
+   .FindByShortName("systemTemp");
+// To avoid duplicate results only add the ones that don't already have members in the inputs CxList
+inputs.Add(directorySystemTemps - inputs.GetTargetOfMembers());

 //sanitizers
 CxList deleteMethod = methdInvk.FindByShortName("deleteSync");
```

Dart / Dart__Mobile__Low__Visibility / User__Information__in__Publicly__Accessible__Storage

Code changes

```
---
+++
@@ -1,9 +1,12 @@
-CxList inputs = Find_UI_Inputs();
-inputs.Add(Find_Data_Storage_Inputs());
+if (Find_Firebase_Auth().Count == 0) //Firebase authentication is the sanitizer for cloud storage. Public accessibility becomes private.
+{
+   CxList inputs = Find_UI_Inputs();
+   inputs.Add(Find_Data_Storage_Inputs());

-CxList sanitizers = Find_Encryption();
-sanitizers.Add(Find_Hashing());
+   CxList sanitizers = Find_Encryption();
+   sanitizers.Add(Find_Hashing());

-CxList sinks = Find_Firebase_Cloud_Sinks();
+   CxList sinks = Find_Firebase_Cloud_Sinks();

-result = sinks.InfluencedByAndNotSanitized(inputs, sanitizers);
+   result = sinks.InfluencedByAndNotSanitized(inputs, sanitizers);
+}
```

Dart / Dart__Mobile__Medium__Threat / Improper__Certificate__Validation

Code changes

```
---
+++
@@ -2,7 +2,6 @@
 CxList memberAccesses = Find_MemberAccesses();
 CxList returnStmts = Find_ReturnStmt();
 CxList expressions = Find_Expressions();
-CxList ifs = Find_Ifs();
 CxList trueAbsValues = Find_True_Abstract_Value();
 trueAbsValues.Add(Find_BooleanLiteral().FindByShortName("true"));

@@ -19,7 +18,11 @@
```

```
 CxList returnValues = returnStmts.GetByAncs(badCertificateCallbackLambda);

 CxList returnExprs = expressions.GetByAncs(returnValues);


-// Any kind of verification is considered sanitization

-returnExprs -= returnExprs.GetByAncs(ifs);

+// Only verifications to host name (lambda's second parameter) are considered sanitizers

+CxList lambdaHostDecl = Find_ParamDecl().GetParameters(badCertificateCallbackLambda, 1);

+CxList lambdaHostRefs = Find_UnknownReference().FindAllReferences(lambdaHostDecl);

+CxList hostsInConditions = lambdaHostRefs.FindByFathers(Find_Conditions());

+CxList sanitizedIfs = hostsInConditions.GetAncOfType<IfStmt>();

+returnExprs -= returnExprs.GetByAncs(sanitizedIfs);


 result = returnExprs * trueAbsValues;
```

Dart / Dart__Mobile__Medium__Threat / Information__Exposure__Through__Query__String

Code changes

```
---

+++

@@ -1,15 +1,18 @@

 CxList methodInvokes = Find_Methods();

+CxList objectCreations = Find_ObjectCreations();


 //input

 CxList sensitiveInfo = Find_Sensitive_Information();

-CxList uriParse = methodInvokes.FindByMemberAccess("Uri.parse");

+string[] uriMemberAccesses = new string[] {"Uri.parse", "Uri.tryParse", "Uri.replace", "Uri.resolve",

+   "Uri.http", "Uri.https"};

+CxList uriObjects = methodInvokes.FindByMemberAccesses(uriMemberAccesses);

+uriObjects.Add(objectCreations.FindByShortName("Uri"));


 //only sensitive info that influences the URL string are relevant

-CxList relevantSensitiveInfo = sensitiveInfo.DataInfluencingOn(uriParse).GetFirstNodesInPath();

+CxList relevantSensitiveInfo = sensitiveInfo.DataInfluencingOn(uriObjects).GetFirstNodesInPath();


 //sinks

 CxList sinks = Find_Remote_Sinks();

-sinks -= Find_WebSockets_Sinks();


 // Any form of encryption is considered a sanitizer

 CxList sanitizers = Find_Encryption();
```

Dart / Dart__Mobile__Medium__Threat / Third__Party__Keyboards__On__Sensitive__Field

Code changes

```
---

+++

@@ -4,7 +4,7 @@

 if (apps.Count == ExtKeyBoardEnabled.Count)

 {
    //sinks
-   CxList vulnerableTextFields = Find_ObjectCreations().FindByShortNames("TextFormField", "TextField");
+   CxList vulnerableTextFields = Find_Widgets_Inputs();


    //input
    CxList sensitiveInfo = Find_Sensitive_Information();
```

Go / Go_Low_Visibility / Deprecated_API

Code changes

```
---
+++
@@ -1,22 +1,23 @@
-CxList mthds = Find_Methods();
+CxList methods = Find_Methods();
+
 // Packages and methods which have been deprecated upstream
-List < string > deprecatedPkgs = new List <string> {@"""io/ioutil""", @"""crypto/dsa"""};
-List < string > deprecatedMthds = new List <string> {
-       "GetQueuedCompletionStatus",
-       "PostQueuedCompletionStatus",
-       "Syscall",
-       "Syscall6",
-       "Syscall9",
-       "Syscall12",
-       "Syscall15",
-       "Syscall18"};
-string[] deprecatedMembers = new string[] {
-       "net.error.Temporary"
-       };
+List<string> deprecatedPkgs = new List<string> { @"""io/ioutil""", @"""crypto/dsa""" };
+List<string> deprecatedMthds = new List<string>
+{
+   "GetQueuedCompletionStatus", "PostQueuedCompletionStatus",
+   "Syscall", "Syscall6", "Syscall9", "Syscall12", "Syscall15", "Syscall18"
+};
+string[] deprecatedMembers = new string[] { "net.error.Temporary" };

 CxList deprecatedImports = Find_Import().FindByShortNames(deprecatedPkgs);
-CxList deprecatedMethods = mthds.FindByShortNames(deprecatedMthds);
-deprecatedMethods.Add(mthds.FindByMemberAccesses(deprecatedMembers));
+CxList deprecatedMethods = methods.FindByShortNames(deprecatedMthds);
+deprecatedMethods.Add(methods.FindByMemberAccesses(deprecatedMembers));
+
+// Deprecated Gin Web Framework (gin-gonic) APIs
+CxList ginContext = Find_Gin_Gonic_Types(new List<string>() {"gin.Context", "Context"});
+deprecatedMethods.Add(ginContext.GetMembersOfTarget().FindByShortName("BindWith"));
+CxList ginEngine = Find_Gin_Gonic_Types(new List<string>() {"gin.Engine", "Engine"});
+deprecatedMethods.Add(ginEngine.GetMembersOfTarget().FindByShortName("AppEngine"));
```

```
 // Methods and packages can also be deprecated incode by adding a comment starting with Deprecated
 CxList deprecateComments = All.FindByRegexExt(@"// Deprecated");
```

Go / Go_Low_Visibility / Open_Redirect

Code changes

```
---
+++
@@ -1,14 +1,30 @@
+CxList unkRefs = Find_UnknownReference();
+
+// Inputs
 CxList inputs = Find_Remote_Requests();
-CxList unkRefs = Find_UnknownReference();
+
+// Outputs
 CxList httpImport = All.FindByMemberAccess("net/http.*");
-CxList sanitizers = Find_String_Sanitizer();
-sanitizers.Add(Find_General_Sanitize());
 CxList outputs = httpImport.FindByShortNames(new List<string>{"Redirect", "RedirectHandler"});

 // Redirects from Gin framework
-CxList ginContextRefs = Find_Gin_Gonic_Types(new List<string>() {"gin.Context", "Context"});
+CxList ginContextRefs = Find_Gin_Gonic_Types(new List<string>() {"gin.Context", "Context"});
+// c.Redirect(302, param)
 CxList contextRedirects = ginContextRefs.GetMembersOfTarget().FindByShortName("Redirect");
 outputs.Add(contextRedirects);
+// c.Header("Location", param)
+CxList contextHeader = ginContextRefs.GetMembersOfTarget().FindByShortName("Header");
+// c.Writer.Header().Add("Location", param)
+CxList cWriterHeaderAdd = ginContextRefs.GetMembersOfTarget().FindByShortName("Writer")
+   .GetMembersOfTarget().FindByShortName("Header")
+   .GetMembersOfTarget().FindByShortName("Add");
+CxList contextHeadersAdd = All.NewCxList(contextHeader, cWriterHeaderAdd);
+CxList locationStrings = Find_String_Literal().FindByShortName("Location");
+CxList vulnContextHeader = locationStrings.GetParameters(contextHeadersAdd, 0).GetAncOfType<MethodInvokeExpr>();
+outputs.Add(vulnContextHeader);
+
+// Sanitizers
+CxList sanitizers = All.NewCxList(Find_String_Sanitizer(), Find_General_Sanitize(), Find_Open_Redirect_Sanitizers());

 CxList relevantArguments = unkRefs.GetParameters(outputs.FindByShortName("Redirect"), 2);
 relevantArguments.Add(unkRefs.GetParameters(outputs.FindByShortName("RedirectHandler"), 0));
```

Go / Go_Low_Visibility / Race_Condition_In_Cross_Functionality

Code changes

```
---
+++
@@ -29,7 +29,12 @@
 CxList varsToExclude = Find_ParamDecl().GetByAncs(parallelMethods);
 varsToExclude -= varsToExclude.FindByType("Pointer");
 varsToExclude -= vars.FindByShortName("this").GetByAncs(parallelMethods);
-varsToExclude.Add(parallelMethodsDecl, sanitizedInvokeType, varsFromImports);
+// Get Gin context sanitized vars "cCp := c.Copy()" outside parallel method
+CxList ginContext = Find_Gin_Gonic_Types(new List<string>() {"gin.Context", "Context"});
+CxList contextCopy = ginContext.GetMembersOfTarget().FindByShortName("Copy").GetAssignee();
+contextCopy -= contextCopy.GetByAncs(parallelMethods);
+
+varsToExclude.Add(parallelMethodsDecl, sanitizedInvokeType, varsFromImports, contextCopy);
 varsToExclude = vars.FindAllReferences(varsToExclude);

 CxList rightMembers = vars.GetRightmostMember().FindByType<MethodInvokeExpr>().FindByTypes(sanitizedVars);
```

Go / Go_Medium_Threat / Cleartext_Transmission_Of_Sensitive_Information

Code changes

```
---
+++
@@ -18,8 +18,7 @@

 // gin-gonic
 CxList engines = Find_Gin_Gonic_Types(new List<string>() {"Engine"});
-httpHandleFunc.Add(engines.GetMembersOfTarget().FindByShortNames("GET", "POST", "PUT", "PATCH", "DELETE", "Handle",
-   "HEAD", "OPTIONS", "Any"));
+httpHandleFunc.Add(engines.GetMembersOfTarget());

 CxList httpHandleFuncParams = All.GetParameters(httpHandleFunc);
 CxList paramWithSensitiveDataList = httpHandleFuncParams * methodDeclsAndLambdasWithPiRefs;
```

Go / Go_Medium_Threat / Reflected_Absolute_Path_Traversal

Code changes

```
---
+++
@@ -1,9 +1,13 @@
+// Find relevant CxLists helpers
+CxList methods = Find_Methods();
+CxList unkRefs = Find_UnknownReference();
 CxList storedInputs = All.NewCxList();
 storedInputs.Add(Find_Read(), Find_DB_Out());

+// Find inputs, sinks(outputs) and sanitzers
 CxList inputs = Find_Inputs() - storedInputs;
 inputs.Add(
-   Find_UnknownReference().GetParameters(Find_Methods().FindByMemberAccess("filepath.WalkFunc"), 0),
+   unkRefs.GetParameters(methods.FindByMemberAccess("filepath.WalkFunc"), 0),
    Find_ParamDecl().GetParameters(Find_MethodDecls().FindByShortName("walkFunc", false), 0));
 CxList outputs = Find_Absolute_Path_Sinks();
@@ -13,8 +17,5 @@
 result = inputs.InfluencingOnAndNotSanitized(outputs, sanitizers)
    .ReduceFlowByPragma().ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

-CxList respWriterParameters = Find_TypeRef().FindByName("http.ResponseWriter").GetAncOfType<ParamDecl>();
-CxList respWriters = Find_Methods().FindByMemberAccess("fmt.Fprintf").FindByParameters(
-   Find_UnknownReference().FindAllReferences(respWriterParameters));
-
-result.Add(inputs.GetRightmostMember().InfluencingOn(respWriters));
+CxList othersPaths = Find_PathTraversal(inputs, outputs, sanitizers);
+result.Add(othersPaths);
```

Go / Go_Medium_Threat / Reflected_Relative_Path_Traversal

Code changes

```
---
+++
@@ -1,6 +1,8 @@
+// Find relevant CxLists helpers
 CxList storedInputs = All.NewCxList();
 storedInputs.Add(Find_Read(), Find_DB_Out());

+// Find inputs, sinks(outputs) and sanitzers
 CxList inputs = Find_Inputs() - storedInputs;
 inputs.Add(
    Find_UnknownReference().GetParameters(Find_Methods().FindByMemberAccess("filepath.WalkFunc"), 0),
@@ -12,3 +14,6 @@

 result = inputs.InfluencingOnAndNotSanitized(outputs, sanitizers)
                .ReduceFlowByPragma().ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
+
+CxList othersPaths = Find_PathTraversal(inputs, outputs, sanitizers);
+result.Add(othersPaths);
```

Java / Java_Android / Client_Side_Injection

Code changes

```
---
+++
@@ -16,33 +16,31 @@
 CxList methodsOfClassExtendContentProviver = methodDeclaration.GetByAncs(classExtendsContentProvider);
 CxList contentProviderFirstParameter = All.GetParameters(methodsOfClassExtendContentProviver, 0);

-CxList contentResolverMembers = All.FindByMemberAccess("ContentResolver.*");
-CxList contentProviderMembers = All.FindByMemberAccess("ContentProvider.*");
-
-CxList contentQuery = contentResolverMembers.FindByShortNames("query", "update");
-contentQuery.Add(contentProviderMembers.FindByShortNames("query", "update"),
-               ContentProvider.FindByShortNames(new List<string> {"query","update"}));
```

```
+CxList contentQuery = methods.FindByMemberAccesses(new string [] {"ContentProvider", "ContentResolver"},
+   new string []{"query", "update"});
+contentQuery.Add(ContentProvider.FindByShortNames(new string []{"query", "update"}));
 db.Add(All.GetParameters(contentQuery, 2));

-CxList contentDelete = contentResolverMembers.FindByShortName("delete");
-contentDelete.Add(contentProviderMembers.FindByShortName("delete"), ContentProvider.FindByShortName("delete"));
+CxList contentDelete = methods.FindByMemberAccesses(new string [] {"ContentProvider", "ContentResolver"},
+   new string []{"delete"});
+contentDelete.Add(ContentProvider.FindByShortName("delete"));
 db.Add(All.GetParameters(contentDelete, 1));

 CxList inputs = All.NewCxList();

-CxList getSharedPref = All.FindByMemberAccess("Context.getSharedPreferences").GetMembersOfTarget();
-CxList sharedPrefString = All.FindByMemberAccesses(new string[]{"SharedPreferences.getString"
-                                                                  , "SharedPreferences.getStringSet"});
-sharedPrefString.Add(getSharedPref.FindByShortNames(new List<string> {"getString","getStringSet"}));
+CxList getSharedPref = methods.FindByMemberAccess("Context.getSharedPreferences").GetMembersOfTarget();
+CxList sharedPrefString = methods.FindByMemberAccesses(new string[]{"SharedPreferences.getString"
+   , "SharedPreferences.getStringSet"});
+sharedPrefString.Add(getSharedPref.FindByShortNames(new string []{"getString","getStringSet"}));

 CxList textWatcher = create.GetParameters(create.FindByShortName("TextWatcher"));
 CxList textChanged = methodDeclaration.GetByAncs(classes.FindByShortName(textWatcher)).FindByShortName("*TextChanged");
 inputs.Add(Find_Read(),
-           sharedPrefString,
-           methods.FindByMemberAccesses(new string[]{"SmsMessage.get*", "Folder.get*",
-           "EditText.getText", "TextView.getText"}),
-           contentProviderFirstParameter,
-           All.GetParameters(textChanged, 0));
+   sharedPrefString,
+   methods.FindByMemberAccesses(new string[]{"SmsMessage.get*", "Folder.get*",
+   "EditText.getText", "TextView.getText"}),
+   contentProviderFirstParameter,
+   All.GetParameters(textChanged, 0));

 CxList strings = Find_Strings();
```

Java / Java_Android / Client_Side_ReDoS

Code changes

```
---
+++
@@ -1,5 +1,6 @@
 // Query Client_Side_DoS
 // ---------------------
+CxList methods = Find_Methods();
```

```
 CxList evilStringsForReplace = Find_Evil_Strings_For_Replace();
 CxList match = Find_Match();
 CxList inputs = Find_Interactive_Inputs();
@@ -19,29 +20,24 @@
    Find_ReDoS(evilStringsForReplace, regexInSecondParam, 1, false, false));

 // Find all regex commands
-CxList regex = All.FindByMemberAccess("Pattern.compile");
+CxList regex = methods.FindByMemberAccess("Pattern.compile");
 // Find regex commands that are influenced by evil strings
 CxList activeEvilRegexes = evilStrings.DataInfluencingOn(regex);
 // Find all matches/splits of regexes
-CxList matches = All.FindByMemberAccess("Matcher.matches");
+CxList matches = methods.FindByMemberAccess("Matcher.matches");
 matches = matches.DataInfluencedBy(inputs);
-CxList split = All.FindByMemberAccess("Pattern.split");
+CxList split = methods.FindByMemberAccess("Pattern.split");
 split = split.DataInfluencedBy(inputs);

 // Find relevant matches
-CxList matchesSplit = All.NewCxList();
-matchesSplit.Add(matches, split);
+CxList matchesSplit = All.NewCxList(matches, split);

 result.Add(activeEvilRegexes.DataInfluencingOn(matchesSplit));

 // ReDoS from Regex Injection
-CxList methods = Find_Methods();
+CxList inputsMember = All.NewCxList(inputs, methods.FindByMemberAccesses(new string [] {"SmsMessage.get*", "Folder.get*"}));

-CxList inputsMember = All.NewCxList();
-inputsMember.Add(inputs, methods.FindByMemberAccesses(new string [] {"SmsMessage.get*", "Folder.get*"}));
-
-CxList matchFirstParam = All.NewCxList();
-matchFirstParam.Add(match, regexInFirstParam, All.FindByName("Pattern.compile"));
+CxList matchFirstParam = All.NewCxList(match, regexInFirstParam, All.FindByName("Pattern.compile"));

 result.Add(Find_ReDoS(inputsMember, matchFirstParam, 0, false, false),
            Find_ReDoS(regexInSecondParam, 1, false));
```

Java / Java_Android / Copy_Paste_Buffer_Caching

Code changes

```
---
+++
@@ -1,15 +1,12 @@
 CxList editTexts = All.FindByType("EditText");

-CxList password = All.NewCxList();
```

```
-password.Add(Find_All_Passwords(), Find_Personal_Info(), Find_Password_Info());
+CxList password = All.NewCxList(Find_All_Passwords(), Find_Personal_Info(), Find_Password_Info());
 CxList sensitiveEditTexts = editTexts * password;

 //Remove EditText which set TYPE_TEXT_VARIATION_PASSWORD.
 CxList memberAccesses = Find_MemberAccess();
-CxList binaryExpression = Find_BinaryExpr();

-CxList potentialParameters = All.NewCxList();
-potentialParameters.Add(memberAccesses, binaryExpression);
+CxList potentialParameters = All.NewCxList(memberAccesses, Find_BinaryExpr());

 CxList passwordField = memberAccesses.FindByShortName("TYPE_TEXT_VARIATION_PASSWORD");
 CxList setInputType = sensitiveEditTexts.GetMembersOfTarget().FindByShortName("setInputType");
@@ -24,21 +21,18 @@

 //Remove cache clearing methods when exiting the application or when going to the backgorund
 CxList classes = Find_Class_Decl();
-List<string> allActivities = new List<string> {"PreferenceActivity", "Activity", "FragmentActivity", "ListActivity",
-       "AppCompatActivity"};
-CxList activity = classes.FindByShortName("*Activity");
-foreach(string activityType in allActivities)
-{
-   activity.Add(classes.InheritsFrom(activityType));
-}
+string [] allActivities = new string []{"PreferenceActivity", "Activity", "FragmentActivity",
+   "ListActivity","AppCompatActivity"};
+
+CxList activity = All.NewCxList(
+   classes.FindByShortName("*Activity"),
+   classes.InheritsFrom(allActivities));

 CxList activityMethods = Find_MethodDeclaration().GetByAncs(activity);
-CxList closeMethods = activityMethods.FindByShortNames(new List<string> {"onDestroy", "onPause", "onStop"});
-CxList methods = Find_Methods();
-CxList setPrimaryClip = methods.FindByMemberAccess("ClipboardManager.setPrimaryClip");
+CxList closeMethods = activityMethods.FindByShortNames(new string[] {"onDestroy", "onPause", "onStop"});
+CxList setPrimaryClip = Find_Methods().FindByMemberAccess("ClipboardManager.setPrimaryClip");

-CxList relevantItems = All.NewCxList(); ;
-relevantItems.Add(sensitiveEditTexts, closeMethods);
+CxList relevantItems = All.NewCxList(sensitiveEditTexts, closeMethods);

 foreach(CxList activityClass in activity)
 {
```

Java / Java_Android / Failure_To_Implement_Least_Privilege

Code changes

```diff
---
+++
@@ -8,12 +8,11 @@
 CxList androidPermission = settings.FindByShortName("*android.permission.*", false);
 // Application Requiered Network access but not uses it
 CxList permissionInternet = androidPermission.FindByShortName("*android.permission.INTERNET*", false);
-CxList usingNetwork = All.NewCxList();
-usingNetwork.Add(
+CxList usingNetwork = All.NewCxList(
    typeRef.FindByTypes(new string[]{ "HttpClient", "OkHttpClient" }),
    methods.FindByMemberAccess("URL.openConnection", false),
    methods.FindByMemberAccesses(new string[]{ "Connector.open", "Transport.send", "Retrofit.create" }),
-   All.FindByShortNames(new List<string>{ "HTTPConnection", "HttpGet"}),
+   All.FindByShortNames(new string[]{ "HTTPConnection", "HttpGet"}),
    All.FindByType("Socket"));

 if ((permissionInternet.Count > 0) && (usingNetwork.Count == 0)){
@@ -21,7 +20,7 @@
 }

 // Application Requiered SMS access but not uses it
-CxList permissionSMS = androidPermission.FindByShortNames(new List<string>{
+CxList permissionSMS = androidPermission.FindByShortNames(new string[]{
        "*android.permission.SEND_SMS*",
        "*android.permission.RECEIVE_SMS*" }, false);

@@ -53,7 +52,7 @@
 CxList actionCall = All.FindByMemberAccess("Intent.ACTION_CALL");
 CxList intentToCall = actionCall.GetParameters(createIntent);

-CxList permissionPhone = androidPermission.FindByShortNames(new List<string> {
+CxList permissionPhone = androidPermission.FindByShortNames(new string[] {
        "*android.permission.READ_PHONE_STATE*",
        "*android.permission.MODIFY_PHONE_STATE*",
        "*android.permission.PROCESS_OUTGOING_CALLS*",
@@ -67,13 +66,12 @@
 }

 // Application Requiered GPS access but does not use it
-CxList permissionGPS = androidPermission.FindByShortNames(new List<string> {
+CxList permissionGPS = androidPermission.FindByShortNames(new string[] {
        "*android.permission.ACCESS_FINE_LOCATION*",
        "*android.permission.ACCESS_COARSE_LOCATION*"}, false);

-CxList usingGPS = All.NewCxList();
-usingGPS.Add(
-   All.FindByShortNames(new List<string>{ "*LocationManager*", "*LocationListener*" }, false),
+CxList usingGPS = All.NewCxList(
+   All.FindByShortNames(new string[]{ "*LocationManager*", "*LocationListener*" }, false),
    All.FindByMemberAccesses(new string[]{ "*MapView.*", "*MapActivity*.*", "*GeoPoint.*", "*MyLocationOverlay.*" }, false));

 if ((permissionGPS.Count > 0) && (usingGPS.Count == 0)){
@@ -82,16 +80,13 @@

 // Application Requiered Contacts access but not uses it

-CxList permissionContacts = All.NewCxList();
-permissionContacts.Add(androidPermission.FindByShortNames(new List<string>{
-       "*android.permission.READ_CONTACTS*",
-       "*android.permission.WRITE_CONTACTS*" }, false));
+CxList permissionContacts = All.NewCxList(androidPermission.FindByShortNames(new string[]{
+   "*android.permission.READ_CONTACTS*",
+   "*android.permission.WRITE_CONTACTS*" }, false));

-CxList usingContacts = All.NewCxList();
-usingContacts.Add(All.FindByShortName("*ContactsContract*"),
+CxList usingContacts = All.NewCxList(All.FindByShortName("*ContactsContract*"),
    All.FindByNames(new string [] {"*android.provider.CallLog*","*Contacts.People*",
-                                    "*Contacts.Phones*", "*Contacts.Photos*", "*Contacts.Organizations*"}));
-
+   "*Contacts.Phones*", "*Contacts.Photos*", "*Contacts.Organizations*"}));

 if ((permissionContacts.Count > 0) && (usingContacts.Count == 0))
 {
@@ -107,14 +102,14 @@
    androidPermission.FindByShortName("*android.permission.WRITE_EXTERNAL_STORAGE*", false);

 CxList usingExternalStorage = All.FindByShortName(@"*/sdcard/*");
-usingExternalStorage.Add(All.FindByMemberAccess("Environment.getExternalStorageDirectory"));
+usingExternalStorage.Add(methods.FindByMemberAccess("Environment.getExternalStorageDirectory"));

 if ((permissionExternalStorage.Count > 0) && (usingExternalStorage.Count == 0)){
    result.Add(permissionExternalStorage);
 }
 // Application Requiered access to use camera but not uses it
 CxList permissionCamera = androidPermission.FindByShortName("*android.permission.CAMERA*", false);
-CxList usingCamera = All.FindByMemberAccess("*Camera.open*");
+CxList usingCamera = methods.FindByMemberAccess("*Camera.open*");
 if ((permissionCamera.Count > 0) && (usingCamera.Count == 0))
 {
    result.Add(permissionCamera);
@@ -146,24 +141,23 @@

 // Manage accounts (add / remove / change credentials)
 CxList permissionManageAccounts = androidPermission.FindByName("\"android.permission.MANAGE_ACCOUNTS\"");
-CxList accountManager = All.FindByMemberAccess("AccountManager.*");
```

```
-CxList manageAccounts = All.NewCxList();
-manageAccounts.Add(accountManager.FindByMemberAccesses(new string[]{
-   "AccountManager.addAccount",
-   "AccountManager.removeAccount",
-   "AccountManager.clearPassword",
-   "AccountManager.confirmCredentials",
-   "AccountManager.editProperties",
-   "AccountManager.getAuthTokenByFeatures",
-   "AccountManager.updateCredentials"}));
+CxList manageAccounts = All.NewCxList(
+   methods.FindByMemberAccesses("AccountManager", new string[]{
+   "addAccount",
+   "removeAccount",
+   "clearPassword",
+   "confirmCredentials",
+   "editProperties",
+   "getAuthTokenByFeatures",
+   "updateCredentials"}));

 if(permissionManageAccounts.Count > 0 && manageAccounts.Count == 0)
    result.Add(permissionManageAccounts);

 // Change configuration
 CxList permissionChangeConfig = androidPermission.FindByName("\"android.permission.CHANGE_CONFIGURATION\"");
-CxList changesConfig = All.FindByMemberAccess("Configuration.set*");
+CxList changesConfig = methods.FindByMemberAccess("Configuration.set*");
 if(permissionChangeConfig.Count > 0 && changesConfig.Count == 0)
 {
    result.Add(permissionChangeConfig);
```

Java / Java_Android / Implicit_Intent_With_Read_Write_Permissions

Code changes

```
---
+++
@@ -3,9 +3,8 @@
 // FLAG_GRANT_WRITE_URI_PERMISSION permissions.

 CxList methods = Find_Methods();
-CxList intent = methods.FindByMemberAccess("Intent.*", false);

-CxList intentFlags = intent.FindByMemberAccesses(new string [] {"Intent.setFlags", "Intent.addFlags"});
+CxList intentFlags = methods.FindByMemberAccesses(new string [] {"Intent.setFlags", "Intent.addFlags"});

 CxList flagsReadWrite = All.FindByMemberAccesses(new string [] {"Intent.FLAG_GRANT_READ_URI_PERMISSION",
    "Intent.FLAG_GRANT_WRITE_URI_PERMISSION"});
@@ -14,12 +13,12 @@
```

```
 // The sanitizer is setting an explicit component / class.
 // The vulnerability is only for an implicit intent.
-CxList sanitize = intent.FindByMemberAccesses(new string [] {"Intent.setComponent", "Intent.setClass"});
+CxList sanitize = methods.FindByMemberAccesses(new string [] {"Intent.setComponent", "Intent.setClass"});

 sanitize = sanitize.GetTargetOfMembers();



-CxList startActivity = methods.FindByShortNames(new List<string> {
+CxList startActivity = methods.FindByShortNames(new string[] {
        "startActivity",
        "startService",
        "startActivities"});
```

Java / Java_Android / Improper_Verification_Of_Intent_By_Broadcast_Receiver

Code changes

```
---
+++
@@ -1,15 +1,14 @@
 //Query Improper_Verification_Of_Intent_By_Broadcast_Receiver
 //-----------------------------------------------------------
 //This query finds intents which are received by a BroadcastReceiver and their source is not verified
-CxList methodDecl = Find_MethodDecls();
 CxList methods = Find_Methods();
-CxList onReceive = methodDecl.FindByShortName("onReceive");
+CxList onReceive = Find_MethodDecls().FindByShortName("onReceive");
 CxList onReceiveIntent = All.GetParameters(onReceive, 1).FindByType("Intent");

 CxList conditions = Find_Conditions();
 conditions.Add(methods.FindByFathers(conditions).FindByShortName("equals"));

-CxList intentActionCheck = All.FindByMemberAccesses(new string [] {"Intent.getAction",
+CxList intentActionCheck = methods.FindByMemberAccesses(new string [] {"Intent.getAction",
                                                                     "Intent.cloneFilter",
                                                                     "Intent.filterEquals",
                                                                     "Intent.filterHashCode"});
```

Java / Java_Android / Information_Leak_Through_Response_Caching

Code changes

```
---
+++
@@ -1,6 +1,9 @@
 //SAP Improvments new Query for Response cache
 //Check if cache response is installed
-CxList responseCacheInstalled = All.FindByMemberAccess("HttpResponseCache.install");
+CxList methods = Find_Methods();
+CxList parameters = Find_Params();
+
+CxList responseCacheInstalled = methods.FindByMemberAccess("HttpResponseCache.install");
 responseCacheInstalled.Add(All.FindByShortName("android.net.http.HttpResponseCache").GetAncOfType<MethodInvokeExpr>());

 if(responseCacheInstalled != null)
@@ -9,17 +12,17 @@
    if(responseCacheInstalled.GetAncOfType<MethodDecl>().FindByShortName("onCreate") != null)
    {
        //Preparing source and sink
-       CxList HttpRequest = All.FindByMemberAccess("URL.openConnection");
-       CxList HttpResponse = All.FindByMemberAccess("HttpURLConnection.getInputStream");
+       CxList HttpRequest = methods.FindByMemberAccess("URL.openConnection");
+       CxList HttpResponse = methods.FindByMemberAccess("HttpURLConnection.getInputStream");
        //Adding sanitizers
        CxList IgnorCacheSanitizer = All.NewCxList();
-       CxList IgnorCacheSet = All.FindByMemberAccess("HttpURLConnection.setUseCaches");
-       CxList IgnorCacheHeader = All.FindByMemberAccess("HttpURLConnection.setRequestProperty");
+       CxList IgnorCacheSet = methods.FindByMemberAccess("HttpURLConnection.setUseCaches");
+       CxList IgnorCacheHeader = methods.FindByMemberAccess("HttpURLConnection.setRequestProperty");

-       CxList IgnorCacheFalseParameter = All.GetParameters(IgnorCacheSet, 0).FindByType<Param>().FindByShortName("false");
-       CxList IgnorCacheHeaderParam1 = All.GetParameters(IgnorCacheHeader, 0).FindByType<Param>().FindByShortName("\"\"Cache-Control\"\"");
-       CxList IgnorCacheHeaderParam2 = All.GetParameters(IgnorCacheHeader, 1).FindByType<Param>().FindByShortName("\"\"no-cache\"\"");
-       CxList IgnorCacheHeaderParam3 = All.GetParameters(IgnorCacheHeader, 1).FindByType<Param>().FindByShortName("\"\"max-age=0\"\"");
+       CxList IgnorCacheFalseParameter = parameters.GetParameters(IgnorCacheSet, 0).FindByShortName("false");
+       CxList IgnorCacheHeaderParam1 = parameters.GetParameters(IgnorCacheHeader, 0).FindByShortName("\"\"Cache-Control\"\"");
+       CxList IgnorCacheHeaderParam2 = parameters.GetParameters(IgnorCacheHeader, 1).FindByShortName("\"\"no-cache\"\"");
+       CxList IgnorCacheHeaderParam3 = parameters.GetParameters(IgnorCacheHeader, 1).FindByShortName("\"\"max-age=0\"\"");

        if(IgnorCacheFalseParameter.Count > 0)
        {
```

Java / Java_Android / Insecure_Data_Storage

Code changes

```
---
+++
@@ -10,16 +10,13 @@
 // The purpose of the query is to detect any attempt to write information to external storage
 // The encrypted information which is stored on externel storage will be detected as well

-CxList strings = Find_Strings();
-
-CxList sd = strings.FindByName(@"*/sdcard/*", false);
-sd.Add(All.FindByMemberAccesses(new string[]{
+CxList sd = Find_Strings().FindByName(@"*/sdcard/*", false);
+sd.Add(Find_Methods().FindByMemberAccesses(new string[]{
    "Environment.getExternalStorageDirectory",
    "Environment.getExternalStoragePublicDirectory",
    "Context.getExternalCacheDir",
    "Context.getExternalFilesDir"}));


-CxList allWrites = Find_Write();

-allWrites.Add(Find_FileSystem_Write());

+CxList allWrites = All.NewCxList(Find_Write(), Find_FileSystem_Write());


 result = allWrites.DataInfluencedBy(sd);
```

Java / Java_Android / Insecure_Data_Storage_Usage

Code changes

```
---

+++

@@ -3,11 +3,9 @@

 //Query find All using of External Storage.


 //Find Using of External Storage for File

-CxList CreateOfObject = Find_Object_Create();

-CxList FileObjects = CreateOfObject.FindByShortName("File");

+CxList FileObjects = Find_Object_Create().FindByShortName("File");


-CxList strings = Find_Strings();

-CxList sd = strings.FindByName(@"*/sdcard/*", false);

+CxList sd = Find_Strings().FindByName(@"*/sdcard/*", false);


 CxList MethodInvoke = Find_Methods();

 CxList ExternalStorage = MethodInvoke.FindByMemberAccesses(new string [] {"Environment.getExternalStorageDirectory",
```

Java / Java_Android / Insecure_WebView_Usage

Code changes

```
---

+++

@@ -8,23 +8,18 @@

 ///////////////////////////////////////////////////////////////////////////////////////////


 CxList methods = Find_Methods();

-CxList allParams = Find_Params();

 CxList members = Find_MemberAccess();

 CxList unknownRef = Find_UnknownReference();

 CxList boolean = Find_BooleanLiteral();


-CxList urAndMembers = All.NewCxList();

-urAndMembers.Add(unknownRef, members);

+CxList urAndMembers = All.NewCxList(unknownRef, members);


 CxList webViews = All.FindByType("WebView");

 webViews.Add(urAndMembers.FindByType(Find_Class_Decl().InheritsFrom("WebView")));


 CxList webSettings = webViews.GetMembersOfTarget().FindByShortName("getSettings");
```

```
-CxList membersUnknownRef = All.NewCxList();
-membersUnknownRef.Add(members, unknownRef);
-
-webSettings.Add(membersUnknownRef.FindAllReferences(webSettings.GetAssignee()));
+webSettings.Add(urAndMembers.FindAllReferences(webSettings.GetAssignee()));
 CxList webSettingsMethods = webSettings.GetMembersOfTarget();

 ////WebSettings.setJavaScriptEnabled
@@ -55,8 +50,7 @@
    // string variables with "file://" strings :
    CxList vars = urAndMembers.FindByAbstractValues(fileUrls);

-   CxList fileUrlsVars = All.NewCxList();
-   fileUrlsVars.Add(fileUrls, vars);
+   CxList fileUrlsVars = All.NewCxList(fileUrls, vars);

     // string vars that check start with("file://") :
    CxList fileSanitizers = methods.FindByMemberAccess("String.startsWith").FindByParameters(fileUrlsVars).GetTargetOfMembers();
@@ -81,5 +75,5 @@

 // Find WebSettings.setPluginState(PluginState.ON) or
 // WebSettings.setPluginState(PluginState.ON_DEMAND)
-CxList pluginParameters = allParams.GetParameters(pluginState).FindByShortName("ON*");
+CxList pluginParameters = Find_Params().GetParameters(pluginState).FindByShortName("ON*");
 result.Add(pluginState.FindByParameters(pluginParameters));
```

Java / Java_Android / Insufficient_Application_Layer_Protect

Code changes

```
---
+++
@@ -15,7 +15,6 @@
    //The block below finds access to the network over HTTP and not HTTPS
    CxList pureHTTP = Find_Pure_http();

-   CxList write = Find_Write();
-   write.Add(Find_Request());
+   CxList write = All.NewCxList(Find_Write(), Find_Request());
    result = write.DataInfluencedBy(pureHTTP);
 }
```

Java / Java_Android / Insufficient_Sensitive_Application_Layer

Code changes

```
---
+++
@@ -14,41 +14,37 @@

 if(!isSanitized)
 {
+   CxList methods = Find_Methods();
    //The block below finds access to the network over HTTP and not HTTPS
-   CxList pureHTTP = Find_Pure_http();
-   pureHTTP.Add(All.FindByType("HttpURLConnection"));
+   CxList pureHTTP = All.NewCxList(Find_Pure_http(), All.FindByType("HttpURLConnection"));

-
    // Find outputs that performed over HTTP
-   CxList write = Find_Write();
+   CxList write = All.NewCxList(Find_Write(), Find_Request());

-   write.Add(Find_Request());
    CxList outInfluencedByHttp = write * write.DataInfluencedBy(pureHTTP);

    //support HTTPClient
-   CxList httpClient = All.FindByMemberAccess("*HttpClient.execute");
+   CxList httpClient = methods.FindByMemberAccess("*HttpClient.execute");
    CxList strings = base.Find_Strings();
-   CxList sslSanitizers = All.NewCxList();
-   sslSanitizers.Add(All.FindByShortName("ssl*", false));
-   sslSanitizers.Add(strings.FindByShortName("https://*", false));
+   CxList sslSanitizers = All.NewCxList(
+       All.FindByShortName("ssl*", false),
+       strings.FindByShortName("https://*", false));
+
    CxList sslSanitized = httpClient.DataInfluencedBy(sslSanitizers);
    CxList nonSanitizedClient = httpClient - sslSanitized;
    outInfluencedByHttp.Add(nonSanitizedClient);

    //support OKHttpClient
-   CxList okHttpClient = All.FindByMemberAccess("OkHttpClient.newCall");
+   CxList okHttpClient = methods.FindByMemberAccess("OkHttpClient.newCall");

-   CxList tlsSanitization = All.FindByMemberAccesses(new string [] {"ConnectionSpec.MODERN_TLS",
+   CxList tlsSanitization = methods.FindByMemberAccesses(new string [] {"ConnectionSpec.MODERN_TLS",
                                                                 "ConnectionSpec.COMPATIBLE_TLS"});
    CxList tlsSanitized = okHttpClient.DataInfluencedBy(tlsSanitization);
    CxList notSanitizedOkHttpClient = okHttpClient - tlsSanitized;
    outInfluencedByHttp.Add(notSanitizedOkHttpClient);
-
-

    // Find all paths that include sensitive (personal) data and outstreamed over HTTP
-   CxList sensitiveInfo = All.NewCxList();
-   sensitiveInfo.Add(Find_Personal_Info(), Find_Password_Info());
+   CxList sensitiveInfo = All.NewCxList(Find_Personal_Info(), Find_Password_Info());
    CxList pathResult = sensitiveInfo.DataInfluencingOn(outInfluencedByHttp);
    result = pathResult.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
    result.Add(Find_Volley_Insufficient_Sensitive_Transport_Layer());
```

Java / Java_Android / Keyboard_Cache_Information_Leak

Code changes

```
---
+++
@@ -26,16 +26,13 @@

 //Look inside Java files
 CxList editTexts = All.FindByType(EditTextWidget);
-CxList password = All.NewCxList();
-password.Add(Find_All_Passwords(), Find_Personal_Info(), Find_Password_Info());
+CxList password = All.NewCxList(Find_All_Passwords(), Find_Personal_Info(), Find_Password_Info());
 CxList passwordsEditText = editTexts * password;
 CxList inputTypeMethods = passwordsEditText.GetMembersOfTarget().FindByShortName("setInputType");
 CxList memberAccesses = Find_MemberAccess();
-CxList binaryExpression = Find_BinaryExpr();
-CxList potentialParameters = All.NewCxList();
-potentialParameters.Add(memberAccesses, binaryExpression);
+CxList potentialParameters = All.NewCxList(memberAccesses, Find_BinaryExpr());

-CxList cacheBlocked = memberAccesses.FindByShortNames(new List<string>{"TYPE_TEXT_FLAG_NO_SUGGESTIONS",
+CxList cacheBlocked = memberAccesses.FindByShortNames(new string[]{"TYPE_TEXT_FLAG_NO_SUGGESTIONS",
        "TYPE_TEXT_VARIATION_VISIBLE_PASSWORD"});
 CxList passwordField = memberAccesses.FindByShortName("TYPE_TEXT_VARIATION_PASSWORD");
```

Java / Java_Android / Non_Encrypted_Data_Storage

Code changes

```
---
+++
@@ -1,22 +1,23 @@
 //Non_Encrypted_Data_Storage
 //---------------------------
 //This query finds non-encrypted data saved to shared storage resources
+CxList methods = Find_Methods();

-CxList outputs = All.FindByMemberAccesses(new string [] {"Editor.putString", "Editor.putStringSet", "OutputStream.write"});
+CxList outputs = methods.FindByMemberAccesses(new string [] {"Editor.putString", "Editor.putStringSet",
+   "OutputStream.write"});

-CxList sharedPreferencessEditor = All.FindByMemberAccess("Context.getSharedPreferences").GetMembersOfTarget().FindByShortName("edit");
-sharedPreferencessEditor.Add(All.FindByMemberAccess("SharedPreferences.edit"));
-
+CxList sharedPreferencessEditor = methods.FindByMemberAccess("Context.getSharedPreferences")
+   .GetMembersOfTarget().FindByShortName("edit");
+sharedPreferencessEditor.Add(methods.FindByMemberAccess("SharedPreferences.edit"));
```

```
 sharedPreferencessEditor = sharedPreferencessEditor.GetMembersOfTarget();

 outputs.Add(sharedPreferencessEditor.FindByShortNames(new string [] {"putString", "putStringSet"}));

 CxList sanitizers = All.GetParameters(outputs, 0);
-sanitizers -= All.GetParameters(All.FindByMemberAccess("OutputStream.write"), 0);// new
+sanitizers -= All.GetParameters(methods.FindByMemberAccess("OutputStream.write"), 0);// new

-CxList cryptography = Find_Encrypt();
-cryptography.Add(Find_HashSanitize());
+CxList cryptography = All.NewCxList(Find_Encrypt(), Find_HashSanitize());

 sanitizers.Add(outputs.GetTargetOfMembers(), cryptography);
```

Java / Java_Android / No_Installer_Verification_Implemented

Code changes

```
---
+++
@@ -2,15 +2,13 @@
    If so, then it assumes a proper verification is implemented.
    Otherwise, it will alert.
 */
-CxList unknownReferences = Find_UnknownReference();
-CxList installerPackageName = All.FindByMemberAccess("*.getInstallerPackageName");
-CxList influencedReferences = installerPackageName.DataInfluencingOn(unknownReferences).GetLastNodesInPath();
+CxList installerPackageName = Find_Methods().FindByMemberAccess("*.getInstallerPackageName");
+CxList influencedReferences = installerPackageName.DataInfluencingOn(Find_UnknownReference()).GetLastNodesInPath();
 installerPackageName.Add(influencedReferences);
 CxList ifStatement = installerPackageName.GetAncOfType<IfStmt>();
 ifStatement.Add(installerPackageName.GetAncOfType<TernaryExpr>());

 if(ifStatement.Count == 0)
 {
-   CxList androidSettings = Find_Android_Settings();
-   result = androidSettings.FindByMemberAccess("ACTIVITY.ANDROID_NAME");
+   result = Find_Android_Settings().FindByMemberAccess("ACTIVITY.ANDROID_NAME");
 }
```

Java / Java_Android / Passing_Non_Encrypted_Data_Between_Activities

Code changes

```
---
+++
@@ -1,12 +1,13 @@
 //Passing_Non_Encrypted_Data_Between_Activities
 //---------------------------------------------
 //This query finds non-encrypted data transmission between activities via intents
```

```
+CxList methods = Find_Methods();

-CxList outputs = All.FindByMemberAccesses(new string [] {"Intent.putExtra", "Intent.putStringArrayListExtra"});
+CxList outputs = methods.FindByMemberAccesses(new string [] {"Intent.putExtra", "Intent.putStringArrayListExtra"});

 CxList sanitizers = All.GetParameters(outputs, 0);

-CxList extras = All.FindByMemberAccesses(new string [] {"Intent.putExtras", "Intent.replaceExtras"});
+CxList extras = methods.FindByMemberAccesses(new string [] {"Intent.putExtras", "Intent.replaceExtras"});

 outputs.Add(extras);

@@ -17,12 +18,10 @@
                outputs.GetTargetOfMembers(),
                cryptography);

-CxList integers = Find_Integers();
-
 CxList values = Find_UnknownReference();
-values.Add(Find_Methods()); // Add data returned from methods
+values.Add(methods); // Add data returned from methods
 values -= values.GetMembersOfTarget().GetTargetOfMembers();
 values -= values.GetParameters(outputs);
 values -= values.DataInfluencedBy(cryptography) * values;
-values -= integers;
+values -= Find_Integers();
 result = values.InfluencingOnAndNotSanitized(outputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Android / Poor_Authorization_and_Authentication

Code changes

```
---
+++
@@ -8,5 +8,5 @@
 CxList http = All.FindByName(@"*http*");
 CxList findWrite = Find_Write();
 CxList outInfluencedByHttp = findWrite * findWrite.DataInfluencedBy(http);
-CxList deviceIDInfo = All.FindByMemberAccess("TelephonyManager.getDeviceId");
+CxList deviceIDInfo = Find_Methods().FindByMemberAccess("TelephonyManager.getDeviceId");
 result = deviceIDInfo.DataInfluencingOn(outInfluencedByHttp);
```

Java / Java_Android / Side_Channel_Data_Leakage

Code changes

```
---
+++
@@ -6,18 +6,16 @@
 sanitize.Add(Find_HashSanitize());

 //Find Personal Info
```

```
-CxList passwordIdentifier = All.FindByShortName("*password*", false).GetAncOfType<MethodInvokeExpr>().FindByShortName("findViewById");
+CxList passwordIdentifier = All.FindByShortName("*password*", false).GetAncOfType<MethodInvokeExpr>()
+    .FindByShortName("findViewById");
 CxList editText = All.FindByType("EditText");
 CxList passwordView = editText.DataInfluencedBy(passwordIdentifier) * editText;

-CxList personal_info = All.NewCxList();
-personal_info.Add(Find_Personal_Info(), Find_Password_Info(), All.FindAllReferences(passwordView));
+CxList personal_info = All.NewCxList(Find_Personal_Info(), Find_Password_Info(), All.FindAllReferences(passwordView));

-CxList pwdAndPersonalInfo = Find_Passwords();
-pwdAndPersonalInfo.Add(personal_info);
+CxList pwdAndPersonalInfo = All.NewCxList(Find_Passwords(), personal_info);

-CxList inputs = Find_Inputs();
-inputs.Add(Find_DB_Out());
+CxList inputs = All.NewCxList(Find_Inputs(), Find_DB_Out());

 //Personal info and passwords that are inputs
 CxList dataToLeak = pwdAndPersonalInfo * inputs;
```

Java / Java_Android / Use_Of_Implicit_Intent_For_Sensitive_Communication

Code changes

```
---
+++
@@ -5,8 +5,7 @@
 CxList intent = All.FindByType("Intent");
 CxList intentDecl = intent.FindByType<Declarator>();
 CxList intentCtor = intent.FindByType<ObjectCreateExpr>();
-CxList sensitiveInfo = All.NewCxList();
-sensitiveInfo.Add(Find_Personal_Info(), Find_Password_Info());
+CxList sensitiveInfo = All.NewCxList(Find_Personal_Info(), Find_Password_Info());
 CxList methods = Find_Methods();

 CxList intentCtorParam = All.GetParameters(intentCtor, 1);
@@ -19,7 +18,7 @@
 CxList implicitIntentDecl = intentDecl.FindByInitialization(intentCtor - explicitIntentCtor);

 // Find explicit intent
-CxList component = All.FindByMemberAccesses(new string [] {"Intent.setComponent", "Intent.setClass",
+CxList component = methods.FindByMemberAccesses(new string [] {"Intent.setComponent", "Intent.setClass",
    "Intent.setClassName", "Intent.setPackage",
    "Intent.setSelector"});
```

Java / Java_Android / Use_of_Native_Language

Code changes

```
---
+++
@@ -5,7 +5,6 @@
 // The purpose of the query to find if the application uses a native language
 // The buffer overflow in this situation can break the sandbox security approach

-CxList strings = Find_Strings();
-CxList methodsLoadLib = All.FindByMemberAccess("*System.loadLibrary");
-CxList nativelib = strings.FindByShortName("*native*",false);
+CxList methodsLoadLib = Find_Methods().FindByMemberAccess("*System.loadLibrary");
+CxList nativelib = Find_Strings().FindByShortName("*native*",false);
 result =  methodsLoadLib.FindByParameters(nativelib);
```

Java / Java_Android / Use_of_WebView_AddJavascriptInterface

Code changes

```
---
+++
@@ -18,12 +18,9 @@
    }
 }
 else{
-   // Find sdk version in Manifest.xml files
-   CxList strings = Find_Strings();
-   CxList settings = Find_Android_Settings();
-
-   CxList sdkVersionVar = settings.GetByAncs(All.FindByName("MANIFEST.USES_SDK.ANDROID_MINSDKVERSION"));
-   CxList SdkVersionVal = strings.GetByAncs(sdkVersionVar.GetAncOfType<AssignExpr>());
+   // Find sdk version in Manifest.xml files
+   CxList sdkVersionVar = Find_Android_Settings().GetByAncs(All.FindByName("MANIFEST.USES_SDK.ANDROID_MINSDKVERSION"));
+   CxList SdkVersionVal = Find_Strings().GetByAncs(sdkVersionVar.GetAncOfType<AssignExpr>());
    isInt = int.TryParse(SdkVersionVal.GetName(), out sdkVersion);
 }
```

Java / Java_Android / Weak_Encryption

Code changes

```
---
+++
@@ -4,22 +4,17 @@
 // provider encryption default for AES
 // The query looks for use of DES or AES with ECB block encryption
 ////////////////////////////////////////////////////////////
-CxList methods = Find_Methods();
-CxList strings = Find_Strings();
-CxList declarators = Find_Declarators();
 CxList unknowRef = Find_UnknownReference();

-CxList parameters = All.NewCxList();
```

```
-parameters.Add(strings, unknowRef);
-CxList cipherGetInstance = methods.FindByMemberAccess("Cipher.getInstance");
+CxList parameters = All.NewCxList(Find_Strings(), unknowRef);
+CxList cipherGetInstance = Find_Methods().FindByMemberAccess("Cipher.getInstance");
 CxList encryptionAlgorithm = parameters.GetParameters(cipherGetInstance, 0);

-CxList assigner = All.NewCxList();
-assigner.Add(declarators, unknowRef);
+CxList assigner = All.NewCxList(Find_Declarators(), unknowRef);
 encryptionAlgorithm.Add(assigner.FindAllReferences(encryptionAlgorithm).GetAssigner());
 CxList encryptionStrings = encryptionAlgorithm.FindByType<StringLiteral>();

-result = encryptionStrings.FindByShortNames(new List<string>{
+result = encryptionStrings.FindByShortNames(new string[]{
    "AES",
    "AES/ECB*",
    "DES*"
```

Java / Java_Android / WebView_Cache_Information_Leak

Code changes

```
---
+++
@@ -1,11 +1,9 @@
 CxList methods = Find_Methods();
 CxList strings = Find_Strings();
-CxList unknownRef = Find_UnknownReference();
 CxList members = Find_MemberAccess();
 CxList classes = Find_Class_Decl();

-CxList urAndMembers = All.NewCxList();
-urAndMembers.Add(unknownRef, members);
+CxList urAndMembers = All.NewCxList(Find_UnknownReference(), members);

 // Look for deleting all cache files of the app
 CxList WebViewDatabase = All.FindByType("WebViewDatabase");
@@ -17,8 +15,7 @@

 CxList clearApplicationUserData = methods.FindByMemberAccess("*.clearApplicationUserData");

-CxList clearFormMethodsApplication = All.NewCxList();
-clearFormMethodsApplication.Add(clearFormMethods, clearApplicationUserData);
+CxList clearFormMethodsApplication = All.NewCxList(clearFormMethods, clearApplicationUserData);

 if (clearFormMethodsApplication.Count == 0 )
 {
@@ -27,9 +24,8 @@
    webView.Add(urAndMembers.FindByType(classes.InheritsFrom("WebView")));
    CxList webViewMethods = webView.GetMembersOfTarget() * methods;
```

```
-   CxList loginString = strings.FindByShortName("*login*");
-   loginString.Add(strings.FindByShortName("*register*"));
-   CxList loginPage = loginString.FindByShortNames(new List<string> {"*.htm", "*.html", "*.php", "*.asp"}, false);
+   CxList loginString = strings.FindByShortNames("*login*", "*register*");
+   CxList loginPage = loginString.FindByShortNames(new string[] {"*.htm", "*.html", "*.php", "*.asp"}, false);

    CxList loadURLMethods = webViewMethods.FindByShortName("loadUrl");
    CxList loadURLParameter = All.GetParameters(loadURLMethods, 0);
@@ -66,12 +62,11 @@
    CxList activity = classes.FindByType("Activity");
    activity.Add(classes.InheritsFrom("Activity"));
    CxList activityMethods = Find_MethodDeclaration().GetByAncs(activity);
-   CxList closeMethods = activityMethods.FindByShortNames(new List<string> {"onDestroy", "onPause", "onStop"});
+   CxList closeMethods = activityMethods.FindByShortNames(new string[] {"onDestroy", "onPause", "onStop"});

    // look for invoking of WebView.clearCache(true) which clears the RAM and the chache files.
    CxList clearCache = webViewMethods.FindByShortName("clearCache");
-   CxList boolean = Find_BooleanLiteral();
-   CxList valueTrue = boolean.FindByShortName("true");
+   CxList valueTrue = Find_BooleanLiteral().FindByShortName("true");
    CxList trueParam = valueTrue.GetParameters(clearCache);
    CxList clearCacheTrue = clearCache.FindByParameters(trueParam);
```

Java / Java_APISecurity / Java_WebApi_GetApiList

Code changes

```
---
+++
@@ -50,8 +50,12 @@
 }
 */

-// Find costum attributes
+// Finds the necessary general queries
 CxList customAttributes = Find_CustomAttribute();
+CxList typeRefCollections = Find_TypeRefCollection();
+CxList methodDecls = Find_MethodDecls();
+CxList methodInvokes = Find_Methods();
+
 // Find controllers
 CxList allClasses = customAttributes.FindByShortNames("Controller", "RestController", "RequestMapping").
    GetAncOfType<ClassDecl>();
@@ -59,21 +63,63 @@
 List <string> annotationsNames = new List<string>() {
        "RequestMapping", "GetMapping", "PostMapping", "PutMapping", "DeleteMapping", "PatchMapping"};
 // Find all methods
-CxList allMethodDecls = Find_MethodDeclaration().GetByClass(allClasses);
```

```
-// Keep only public methods
+CxList allMethodDecls = methodDecls.GetByClass(allClasses);
+// Keep only public methods and remove static methods
 allMethodDecls = allMethodDecls.FindByFieldAttributes(Modifiers.Public);
+allMethodDecls -= allMethodDecls.FindByFieldAttributes(Modifiers.Static);
 // Kepp only methods with annotations
 CxList methodsWithAnnotations = allMethodDecls * customAttributes.FindByShortNames(annotationsNames).
    GetAncOfType<MethodDecl>();
-// Get list of methods without annotations
-/*CxList methodsNoAnnotation = allMethodDecls - methodsWithAnnotations;

-CxList findValueImport = Find_Import().FilterByDomProperty<Import>(imp => imp.ImportedFilename == "org.springframework.beans.factory.annotation.Value");
+// Retrieve a list of methods without annotations
+// This section is responsible for filtering the 'methodsNoAnnotation' list based on the following criteria:
+// > @PostConstruct, @ExceptionHandler, @Bean, @Scheduled, @InitBinder and @ModelAttribute annotations
+// > Static methods
+// > @Override annotations
+// > @Value annotations
+// > Method Invokes
+CxList methodsNoAnnotation = allMethodDecls - methodsWithAnnotations;
+
+// Step 1: Filtering by @PostConstruct, @ExceptionHandler, @Bean, @Scheduled, @InitBinder and @ModelAttribute annotations
+CxList methodsToRemove = customAttributes.FindByCustomAttribute("PostConstruct").GetFathers();
+methodsToRemove.Add(customAttributes.FindByCustomAttribute("ExceptionHandler").GetFathers());
+methodsToRemove.Add(customAttributes.FindByCustomAttribute("Bean").GetFathers());
+methodsToRemove.Add(customAttributes.FindByCustomAttribute("Scheduled").GetFathers());
+methodsToRemove.Add(customAttributes.FindByCustomAttribute("InitBinder").GetFathers());
+methodsToRemove.Add(customAttributes.FindByCustomAttribute("ModelAttribute").GetFathers());
+methodsNoAnnotation -= methodsToRemove * methodsNoAnnotation;
+
+// Step 2: Filtering by @Value annotation
+CxList findValueImport = Find_Import()
+   .FilterByDomProperty<Import>(imp => imp.ImportedFilename == "org.springframework.beans.factory.annotation.Value");
 if(findValueImport.Count > 0)
 {
    CxList methodsWithValueAnnotation = customAttributes.FindByCustomAttribute("Value").GetFathers() * methodsNoAnnotation;
    methodsNoAnnotation -= methodsWithValueAnnotation;
-}*/
+}

-result.Add(Java_WebApi_MethodsWithAnnotation(methodsWithAnnotations, customAttributes, annotationsNames));
-// Java_WebApi_MethodsNoAnnotation(methodsNoAnnotation, customAttributes, annotationsNames)
+// Step 3: Remove methods that have @Override annotation but no routing annotations
+CxList classOfMethodNoAnnotation = methodsNoAnnotation.GetAncOfType<ClassDecl>();
+CxList overriddenMethods = methodsNoAnnotation.FindByFieldAttributes(Modifiers.Override);
+CxList allExtendedInterfaces = All.FindDefinition(Find_TypeRef()
+   .FindByFathers(typeRefCollections.FindByFathers(classOfMethodNoAnnotation)));
+CxList allOverriddenMethodsInInterface = methodDecls.GetByClass(allExtendedInterfaces)
```

```
+   .FilterByDomProperty<MethodDecl>(method => method.Name == overriddenMethods.FindByShortName(method.Name).GetName());
+CxList interfacesNoAnnotations = allExtendedInterfaces
+   .FilterByDomProperty<InterfaceDecl>(x => x.CustomAttributes.Count == 0);
+if(allExtendedInterfaces.FindByType<ClassDecl>().Count > 0){
+   interfacesNoAnnotations.Add(allExtendedInterfaces
+       .FilterByDomProperty<ClassDecl>(x => x.CustomAttributes.Count == 0));
+}
+CxList overriddenMethodsToRemove = allOverriddenMethodsInInterface
+   .FilterByDomProperty<MethodDecl>(x => x.CustomAttributes.Count == 0
+   || x.CustomAttributes.Any(a => a.Name == "Override")).GetByAncs(interfacesNoAnnotations);
+
+methodsNoAnnotation -= methodsNoAnnotation.FindByShortName(overriddenMethodsToRemove);
+
+// Step 4: Remove methods invokated by methods with annnotations
+CxList methodInvokesToRemove = methodInvokes.FindByShortName(methodsNoAnnotation).GetByAncs(methodsWithAnnotations);
+methodsNoAnnotation -= allMethodDecls.FindByShortName(methodInvokesToRemove);
+
+result.Add(Java_WebApi_MethodsWithAnnotation(methodsWithAnnotations, customAttributes, annotationsNames),
+   Java_WebApi_MethodsNoAnnotation(methodsNoAnnotation, customAttributes, annotationsNames));
```

Java / Java_AWS_Lambda / AWS_Credentials_Leak

Code changes

```
---
+++
@@ -1,4 +1,3 @@
-CxList methodDecls = Find_Methods();
 CxList unknownRefs = Find_UnknownReference();

 //All Outpus
@@ -21,15 +20,11 @@
 CxList credentialsInfo = credentials.GetMembersOfTarget().FindByShortNames(credentialsInfoName);

 //Get Environment Vars(Keys and Token)
-CxList envVars = Find_String_Literal();
-envVars = envVars.FindByShortNames(
+CxList envVars = Find_String_Literal().FindByShortNames(
    "AWS_ACCESS_KEY",
    "AWS_ACCESS_KEY_ID",
    "AWS_SECRET_ACCESS_KEY",
    "AWS_SESSION_TOKEN");

-CxList stystemGetEnv = methodDecls.FindByName("System.getenv");
-stystemGetEnv = stystemGetEnv.InfluencedBy(envVars).GetLastNodesInPath();
-
-result.Add(outputs.InfluencedBy(credentialsInfo));
-result.Add(outputs.InfluencedBy(envVars));
+CxList credInfoAndEnvVars = All.NewCxList(credentialsInfo, envVars);
+result.Add(outputs.InfluencedBy(credInfoAndEnvVars));
```

Java / Java__AWS__Lambda / Hardcoded__AWS__Credentials

Code changes

```
---
+++
@@ -1,12 +1,10 @@
-CxList methods = Find_Methods();
-CxList instantiations = Find_Object_Create();
 CxList strings = Find_String_Literal();

 string [] namesV1 = new string[] {"BasicAWSCredentials","BasicSessionCredentials"};
 string [] namesV2 = new string[] {"AwsBasicCredentials.create","AwsSessionCredentials.create"};

-CxList awsConstructor = instantiations.FindByShortNames(namesV1);
-awsConstructor.Add(methods.FindByMemberAccesses(namesV2));
+CxList awsConstructor = Find_Object_Create().FindByShortNames(namesV1);
+awsConstructor.Add(Find_Methods().FindByMemberAccesses(namesV2));

 CxList keyParams = All.GetParameters(awsConstructor);
```

Java / Java__AWS__Lambda / Permission__Manipulation__in__S3

Code changes

```
---
+++
@@ -17,7 +17,7 @@
 CxList relevantMethods = methods.FindByShortNames(vulnerableMethods);

 //Get all relevant methods influenced by inputs without losing lambda on flow
-CxList relevantMethodsInfluencedByInputs = All.InfluencedBy(inputs) * relevantMethods;
+CxList relevantMethodsInfluencedByInputs = methods.InfluencedBy(inputs) * relevantMethods;

 //Add all possible relevant params
 CxList relevantParams = inputs.InfluencingOn(unknownRefsParamsOut);
```

Java / Java__AWS__Lambda / Race__Condition__Global__Scope

Code changes

```
---
+++
@@ -1,10 +1,7 @@
-CxList globalVar = Find_FieldDecls();
-CxList methods = Find_Methods();
-CxList methodDecls = Find_MethodDecls();
 CxList lambdaHandler = Find_AWSLambda_Handlers();

 //Get methods invoked inside handleRequest
-CxList methodInvokedLambda = methodDecls.FindDefinition(methods.GetByAncs(lambdaHandler));
+CxList methodInvokedLambda = Find_MethodDecls().FindDefinition(Find_Methods().GetByAncs(lambdaHandler));
 lambdaHandler.Add(methodInvokedLambda);
 //Get Assign and PostFix expressions inside Handlers
@@ -17,8 +14,7 @@

 //Find Lefts Assign Expression
 CxList lefts = assignementsInHandler.CxSelectDomProperty<AssignExpr>(a => a.Left);
-refsAssignedInHandler.Add(lefts.FindByType<UnknownReference>());
+refsAssignedInHandler.Add(lefts.FindByTypes(typeof(UnknownReference), typeof(IndexerRef)));
 refsAssignedInHandler.Add(lefts.FindByType<MemberAccess>().GetTargetOfMembers());
-refsAssignedInHandler.Add(lefts.FindByType<IndexerRef>());

-result = refsAssignedInHandler.FindAllReferences(globalVar);
+result = refsAssignedInHandler.FindAllReferences(Find_FieldDecls());
```

Java / Java_Best_Coding_Practice / Access_Specifier_Manipulation

Code changes

```
---
+++
@@ -4,8 +4,7 @@
 //           true (it could be either the first or the second parameter)

 // setAccessible methods
-CxList methods = Find_Methods();
-CxList setAccessible = methods.FindByMemberAccesses(new string[]{
+CxList setAccessible = Find_Methods().FindByMemberAccesses(new string[]{
    "Field.setAccessible",
    "Method.setAccessible",
    "Constructor.setAccessible",
```

Java / Java_Best_Coding_Practice / Call_to_Thread_run

Code changes

```
---
+++
@@ -1,4 +1,2 @@
-CxList baseRefs = Find_BaseRef();
-
-CxList superRun = All.FindByFathers(baseRefs.GetFathers()).FindByShortName("run");
-result = All.FindByMemberAccess("Thread.run") - superRun;
+CxList superRun = All.FindByFathers(Find_BaseRef().GetFathers()).FindByShortName("run");
+result = Find_Methods().FindByMemberAccess("Thread.run") - superRun;
```

Java / Java_Best_Coding_Practice / Catch_NullPointerException

Code changes

```
---
+++
@@ -1,3 +1,2 @@
-CxList Catch = Find_Catch();
 CxList NullPointerException = All.FindByName("NullPointerException");
```

```
-result = NullPointerException.FindByFathers(Catch);

+result = NullPointerException.FindByFathers(Find_Catch());
```

Java / Java_Best_Coding_Practice / clone_Method_Without_super_clone

Code changes

```
---

+++

@@ -1,12 +1,10 @@

 CxList cloneable = All.InheritsFrom("Cloneable");

-CxList methodDecl = Find_MethodDeclaration();

-CxList methods = Find_Methods();


-CxList clone = methodDecl.GetByAncs(cloneable).FindByShortName("clone");

+CxList clone = Find_MethodDeclaration().GetByAncs(cloneable).FindByShortName("clone");


 // super.clone call. Using FindByShortName, "super" is not a standard member.

 // It sould catch 99.9 % of te cases

-CxList superClone = methods.FindByShortName("clone");

+CxList superClone = Find_Methods().FindByShortName("clone");

 // Find the methos that contains the super.finalize

 CxList cloneWithSuperFinalize = superClone.GetAncOfType<MethodDecl>();
```

Java / Java_Best_Coding_Practice / Comparison_of_Classes_By_Name

Code changes

```
---

+++

@@ -13,8 +13,7 @@

 CxList variables = Find_UnknownReference();

 variables.Add(Find_Declarators());

 //We filter our results by looking only for strings or classes

-CxList toRemove = All.NewCxList();

-toRemove.Add(stringTypes, classTypes, All.FindByMemberAccess("*.class").GetTargetOfMembers());

+CxList toRemove = All.NewCxList(stringTypes, classTypes, All.FindByMemberAccess("*.class").GetTargetOfMembers());


 CxList sanitizers = variables - toRemove;
```

Java / Java_Best_Coding_Practice / Critical_Public_Variable_Without_Final_Modifier

Code changes

```
---

+++

@@ -1,11 +1,9 @@

-CxList classDecl = Find_Class_Decl();

-CxList cl = classDecl.InheritsFrom("Applet");

-cl.Add(classDecl.InheritsFrom("JApplet"));

+CxList cl = Find_Class_Decl().InheritsFrom(new string [] {"Applet", "JApplet"});
```

```
 CxList fields = Find_Field_Decl();


 fields = fields.GetByAncs(cl);
 CxList publicFields = fields.FindByFieldAttributes(Checkmarx.Dom.Modifiers.Public);
-CxList staticFields =  fields.FindByFieldAttributes(Checkmarx.Dom.Modifiers.Static);
+CxList staticFields = fields.FindByFieldAttributes(Checkmarx.Dom.Modifiers.Static);


 result = publicFields - staticFields;
```

Java / Java_Best_Coding_Practice / Dead_Code

Code changes

```
---
+++
@@ -1,7 +1,8 @@
-result = base.Find_False_Conditions();
-result.Add(base.Find_Unused_Private_Methods());
-result.Add(base.Find_Code_After_Return());
-result.Add(base.Find_Unreached_Switch_Case());
-result.Add(base.Find_Catch_Block_Of_Empty_Try());
+result = All.NewCxList(
+   base.Find_False_Conditions(),
+   base.Find_Unused_Private_Methods(),
+   base.Find_Code_After_Return(),
+   base.Find_Unreached_Switch_Case(),
+   base.Find_Catch_Block_Of_Empty_Try());


 result -= Find_Catch_In_Views();
```

Java / Java_Best_Coding_Practice / Declaration_of_Throws_for_Generic_Exception

Code changes

```
---
+++
@@ -1,4 +1,4 @@
 // Generic exceptions in java are considered the classes Exception and Throwable.


-var genericExceptions = new List<string>{"Exception", "Throwable"};
+string[] genericExceptions = new string[]{"Exception", "Throwable"};
 result = Find_Throws_Exceptions().FindByShortNames(genericExceptions, false);
```

Java / Java_Best_Coding_Practice / Detection_of_Error_Condition_Without_Action

Code changes

```
---
+++
@@ -1,13 +1,7 @@
 result = Common_Best_Coding_Practice.Detection_of_Error_Condition_Without_Action();


-CxList mkdirs = All.FindByMemberAccess("File.mkdirs");
+CxList mkdirs = Find_Methods().FindByMemberAccess("File.mkdirs");
```

```
 CxList not = mkdirs.GetAncOfType<UnaryExpr>().FindByShortName("Not");
 CxList If = not.GetFathers().FindByType<IfStmt>();
-foreach(CxList curIf in If)
-{
-   IfStmt ifStmt = curIf.TryGetCSharpGraph<IfStmt>();
-   if(ifStmt.TrueStatements.Count == 0)
-   {
-       result.Add(ifStmt.NodeId, ifStmt);
-   }
-}
+
+result.Add(If.FilterByDomProperty<IfStmt>(x => x.TrueStatements.Count == 0));
```

Java / Java_Best_Coding_Practice / Direct_Use_of_Sockets

Code changes

```
---
+++
@@ -1,4 +1,4 @@
 if(All.isWebApplication)
 {
-   result = All.FindByMemberAccess("ServerSocket.accept");
+   result = Find_Methods().FindByMemberAccess("ServerSocket.accept");
 }
```

Java / Java_Best_Coding_Practice / Dynamic_File_Inclusion

Code changes

```
---
+++
@@ -1 +1 @@
-result = All.FindByMemberAccesses(new string [] {"response.include", "response.Import"});
+result = Find_Methods().FindByMemberAccesses(new string [] {"response.include", "response.Import"});
```

Java / Java_Best_Coding_Practice / Dynamic_Set_Of_Null_SecurityManager

Code changes

```
---
+++
@@ -4,7 +4,6 @@
 
 CxList nulls = All.FindByAbstractValue(_ => _ is NullAbstractValue);
 CxList setSecurityManagers = Find_Methods().FindByMemberAccess("System.setSecurityManager");
-CxList setSecurityManNulls = All.NewCxList();
-setSecurityManNulls.Add(setSecurityManagers, nulls);
+CxList setSecurityManNulls = All.NewCxList(setSecurityManagers, nulls);
 CxList sanitizers = base.Find_Same_Value_Sanitizers_Exclude_Sinks_and_Sources(setSecurityManNulls);
 result = nulls.InfluencingOnAndNotSanitized(setSecurityManagers, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceSmallFlow);
```

Java / Java_Best_Coding_Practice / Dynamic_SQL_Queries

Code changes

```diff
---
+++
@@ -18,13 +18,12 @@
 CxList replace = javaDB.FindByShortName("replace");



-CxList javaDBUnkonRef = All.NewCxList();
-javaDBUnkonRef.Add(javaDB.FindByTypes(typeof(UnknownReference), typeof(Declarator)));
+CxList javaDBUnkonRef = All.NewCxList(javaDB.FindByTypes(typeof(UnknownReference), typeof(Declarator)));

 CxList str = javaDBUnkonRef.FindByType("String");
 str.Add(stringMethods,
        // Find strings of type member access (e.g. a.str)
-       All.FindByType("String").FindByType<MemberAccess>(),
+       Find_MemberAccesses().FindByType("String"),
        // Find toString methods
        methods.FindByShortName("toString"));

@@ -43,8 +42,7 @@
 // binary operations whose descendants are strings
 CxList binaryAncOfStr = str.GetAncOfType<BinaryExpr>();

-CxList concat = All.NewCxList();
-concat.Add(binaryAncOfStr, append, replace, str.GetByAncs(binary.GetByAncs(db)));
+CxList concat = All.NewCxList(binaryAncOfStr, append, replace, str.GetByAncs(binary.GetByAncs(db)));

 // Find the '+=' operators
 CxList assignments = Find_AssignExpr();
@@ -58,11 +56,11 @@
 // Add MyBatis _parameter Object vars that are assigned to strings
 concat.Add(Find_MyBatis_Temp_Parameter_Assigned_To_String());

-CxList substring = All.FindByMemberAccess("String.substring");
+CxList substring = methods.FindByMemberAccess("String.substring");

-CxList sanitize = All.NewCxList();
-sanitize.Add(Find_Parameters(), Find_Dead_Code_Contents(),
-           All.FindByMemberAccess("DriverManager.getConnection"),
-           All.GetByAncs(All.GetParameters(substring)));
+CxList sanitize = All.NewCxList(
+   Find_Parameters(), Find_Dead_Code_Contents(),
+   methods.FindByMemberAccess("DriverManager.getConnection"),
+   All.GetByAncs(All.GetParameters(substring)));

 result = db.InfluencedByAndNotSanitized(concat, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Best_Coding_Practice / Empty_Synchronized_Block

Code changes

Java / Java_Best_Coding_Practice / ESAPI_Banned_API

Code changes

```
---
+++
@@ -43,7 +43,17 @@
    "Session.invalidate",
    "Math.Random",
    "File.createTempFile",
-   "RequestDispatcher.forward"};
+   "RequestDispatcher.forward",
+   //More banned apis
+   "java.security",
+   "javax.crypto",
+   "net.URLEncoder",
+   "net.URLDecoder",
+   "Statement.execute",
+   "ServletContext.log",
+   "ServletRequest.getUserPrincipal",
+   "ServletRequest.isUserInRole",
+   "ServletRequest.isSecure"};

 List<string> properties = new List<string>{
        "getProperty",
@@ -80,17 +90,6 @@
        "notifyAll",
        "wait"};

-string[] moreBannedApis = new string[]{
-   "java.security",
-   "javax.crypto",
-   "net.URLEncoder",
-   "net.URLDecoder",
-   "Statement.execute",
-   "ServletContext.log",
-   "ServletRequest.getUserPrincipal",
-   "ServletRequest.isUserInRole",
-   "ServletRequest.isSecure"};
-
 string[] servletResponses = new string[]{
    "ServletResponse.setContentType",
    "ServletResponse.sendRedirect",
@@ -104,7 +103,6 @@
 result.Add(nonJspOutPrints,
    allWithoutPomFile.FindByMemberAccesses(bannedApis),
    allWithoutPomFile.FindByMemberAccess("Properties.*").FindByShortNames(properties),
-   allWithoutPomFile.FindByMemberAccesses(moreBannedApis),
    temp.FindByMemberAccesses(servletResponses));


 // Remove false positives:
@@ -113,15 +111,13 @@
    "EncryptedProperties.getProperty",
    "EncryptedProperties.keySet",
    "EncryptedProperties.load",
-   "EncryptedProperties.store"};
-
-string[] referenceEncryptedProperties = new string[]{
+   "EncryptedProperties.store",
+   //Reference Encrypted Properties
    "ReferenceEncryptedProperties.getProperty",
    "ReferenceEncryptedProperties.keySet",
    "ReferenceEncryptedProperties.load",
-   "ReferenceEncryptedProperties.store"};
-
-string[] defaultEncryptedProperties = new string[]{
+   "ReferenceEncryptedProperties.store",
+   //Default Encrypted Properties
    "DefaultEncryptedProperties.getProperty",
    "DefaultEncryptedProperties.keySet",
    "DefaultEncryptedProperties.load",
@@ -129,5 +125,3 @@
    "PreparedStatement.execute"};


 result -= result.FindByMemberAccesses(encryptedProperties);
-result -= result.FindByMemberAccesses(referenceEncryptedProperties);
-result -= result.FindByMemberAccesses(defaultEncryptedProperties);
```

Java / Java_Best_Coding_Practice / Explicit_Call_to_Finalize

Code changes

```
---
+++
@@ -1,5 +1,3 @@
-CxList AllMethodInvoke = Find_Methods();
-CxList baseRefs = Find_BaseRef();
-CxList superFinalize = All.FindByFathers(baseRefs.GetFathers()).FindByShortName("finalize");
+CxList superFinalize = All.FindByFathers(Find_BaseRef().GetFathers()).FindByShortName("finalize");


-result = AllMethodInvoke.FindByShortName("finalize") - superFinalize;
+result = Find_Methods().FindByShortName("finalize") - superFinalize;
```

Java / Java_Best_Coding_Practice / Exposure_of_Resource_to_Wrong_Sphere

Code changes

```
---
+++
@@ -7,6 +7,5 @@
```

```
 CxList allStaticFields = allPublicFields.FindByFieldAttributes(Modifiers.Static);


 //remove all constant and public static fields
-CxList toRemove = All.NewCxList();
-toRemove.Add(allConstFields, allStaticFields);
+CxList toRemove = All.NewCxList(allConstFields, allStaticFields);
 result = allPublicFields - toRemove;
```

Java / Java_Best_Coding_Practice / finalize_Method_Declared_Public

Code changes

```
---
+++
@@ -1,9 +1,5 @@
 //classes that inherits from Applet
-CxList classDecl = Find_Class_Decl();
-
-CxList classDeclApplet = All.NewCxList();
-classDeclApplet.Add(classDecl.InheritsFrom("Applet"),
-                    classDecl.InheritsFrom("JApplet"));
+CxList classDeclApplet = All.NewCxList(Find_Class_Decl().InheritsFrom(new string []{"Applet", "JApplet"}));


 CxList classAncs = All.GetByAncs(classDeclApplet);
```

Java / Java_Best_Coding_Practice / finalize_Method_Without_super_finalize

Code changes

```
---
+++
@@ -1,13 +1,9 @@
-// Find all method decl and method def
-CxList methodDecl = Find_MethodDeclaration();
-CxList methods = Find_Methods();
-
 // finalize methods
-CxList finalize = methodDecl.FindByShortName("finalize");
+CxList finalize = Find_MethodDeclaration().FindByShortName("finalize");


 // super.finalize call. Using FindByShortName, "super" is not a standard member.
 // It sould catch 99.9 % of te cases
-CxList superFinalize = methods.FindByShortName("finalize");
+CxList superFinalize = Find_Methods().FindByShortName("finalize");
 // Find the methos that contains the super.finalize
 CxList finalizeWithSuperFinalize = superFinalize.GetAncOfType<MethodDecl>();
```

Java / Java_Best_Coding_Practice / GOTO_Statement

Code changes

```diff
---
+++
@@ -1 +1 @@
-result = All.FindByType(typeof(GotoStmt));
+result = Find_Goto();
```

Java / Java_Best_Coding_Practice / Incorrect_Conversion_between_Numeric_Types

Code changes

```diff
---
+++
@@ -65,7 +65,7 @@
 /// Case 2:
 // Find a sqrt affected by input that is casted
 // Make sure that if there is a checkon the sqrt parameter, then it is OK
-CxList sqrt = All.FindByMemberAccess("Math.sqrt").GetByAncs(inCast);
+CxList sqrt = Find_Methods().FindByMemberAccess("Math.sqrt").GetByAncs(inCast);
 CxList sqrtParam = All.GetParameters(sqrt, 0);
 CxList paramInCondition = All.GetByAncs(allConditions).FindAllReferences(sqrtParam);
```

Java / Java_Best_Coding_Practice / Missing_XML_Validation

Code changes

```diff
---
+++
@@ -3,7 +3,7 @@
    "DocumentBuilderFactory",
    "SAXReader"});

-CxList setValidator = All.FindByMemberAccesses(new string [] {"SAXParserFactory.setValidating",
+CxList setValidator = Find_Methods().FindByMemberAccesses(new string [] {"SAXParserFactory.setValidating",
                                                             "DocumentBuilderFactory.setValidating",
                                                             "SAXReader.setValidation"});
```

Java / Java_Best_Coding_Practice / Pages_Without_Global_Error_Handler

Code changes

```diff
---
+++
@@ -1,8 +1,4 @@
-CxList viewDecls = Find_ViewDecls();
-CxList jspCode = Find_Jsp_Code();
-CxList jsfCode = Find_JSF_Code();
-CxList jsfJspCodes = All.NewCxList();
-jsfJspCodes.Add(jspCode, jsfCode);
+CxList jsfJspCodes = All.NewCxList(Find_Jsp_Code(), Find_JSF_Code());

 CxList jspJsfPages = All.GetClass(All.FindByName("*Checkmarx_Class_Init*").FindByType<MethodDecl>());
 jspJsfPages.Add(
@@ -13,7 +9,7 @@

 CxList errorHandledPagesClasses = viewCallsToErrorPages.GetAncOfType<ClassDecl>();

-CxList jspJsfErrorPages = viewDecls.FindByShortName(viewCallsToErrorPages);
+CxList jspJsfErrorPages = Find_ViewDecls().FindByShortName(viewCallsToErrorPages);
 errorHandledPagesClasses.Add(jspJsfErrorPages.GetAncOfType<ClassDecl>());

 CxList errorHandledPages = All.GetClass(All.FindByName("*page.errorPage"));
```

Java / Java_Best_Coding_Practice / Portability_Flaw_In_File_Separator

Code changes

```
---
+++
@@ -8,14 +8,10 @@
 CxList listOfVarDecl = Find_VariableDeclStmt();
 CxList listOfMemberAccess = Find_MemberAccesses();

-CxList allNeededTypes = All.NewCxList();
-allNeededTypes.Add(listOfParams, listOfUnkRef, stringsWithSeparator);
+CxList allNeededTypes = All.NewCxList(listOfParams, listOfUnkRef, stringsWithSeparator);

 CxList obj = Find_FileObjects();
-CxList listOfDeclAndMemberAndUnknown = All.NewCxList();
-listOfDeclAndMemberAndUnknown.Add(listOfUnkRef, listOfMemberAccess, listOfVarDecl);
-
-
+CxList listOfDeclAndMemberAndUnknown = All.NewCxList(listOfUnkRef, listOfMemberAccess, listOfVarDecl);

 CxList variablesOfStringWithSeparator = stringsWithSeparator.GetAssignee();
 //find all references of string with separator(it will find references inside of a setter)
```

Java / Java_Best_Coding_Practice / Potentially_Serializable_Class_With_Sensitive_Data

Code changes

```
---
+++
@@ -1,6 +1,5 @@
 // Find Sensitive data field (non boolean)
-CxList sensitiveData = All.NewCxList();
-sensitiveData.Add(Find_Personal_Info(), Find_Password_Info());
+CxList sensitiveData = All.NewCxList(Find_Personal_Info(), Find_Password_Info());
 sensitiveData = sensitiveData * Find_Field_Decl();
 sensitiveData -= sensitiveData.FindByType("boolean");
 sensitiveData -= sensitiveData.FindByFileName("*.properties");
```

Java / Java_Best_Coding_Practice / Redirect_Without_Exit

Code changes

```
---
+++
@@ -8,7 +8,7 @@
 */

 // Find all the relevant redirects
-CxList sendRedirect = All.FindByMemberAccesses(new string [] {"HttpServletResponse.sendRedirect",
+CxList sendRedirect = Find_Methods().FindByMemberAccesses(new string [] {"HttpServletResponse.sendRedirect",
                                                              "HTTPUtilities.safeSendRedirect"}); //ESAPI
 sendRedirect.Add(All.FindByName("*response.sendRedirect", false));

@@ -28,8 +28,8 @@
 {
    // Find the redirect under the statements (there might be more than one)
    CxList redirect0 = sendRedirect.GetByAncs(statements);
-   CxList redirect = All.NewCxList();
-   redirect.Add(redirect0);
+   CxList redirect = All.NewCxList(redirect0);
+
    foreach (CxList r in redirect0)
    {
        if (r.GetAncOfType<StatementCollection>() != statements)
@@ -37,7 +37,7 @@
            redirect -= r;
        }
    }
-
+
    // If there are various redirects in the block - just take the first one
    if (redirect.Count > 1)
    {
@@ -73,8 +73,7 @@
    // in case of more than one redirect in a block
    int maxRedirectId = -1;

-   CxList relevantRedirect = All.NewCxList();
-   relevantRedirect.Add(redirect);
+   CxList relevantRedirect = All.NewCxList(redirect);
    foreach (CxList rt in redirectThings)
    {
        try
```

Java / Java_Best_Coding_Practice / Reliance_On_Untrusted_Inputs_In_Security_Decision

Code changes

```
---
+++
@@ -16,28 +16,13 @@
 CxList sink = Find_Sink_Of_Security_Decision();
 // search for If conditions (all the elements in the condition)
-CxList conditions = All.NewCxList();
 CxList ifAllCond = Find_Ifs();
-foreach(CxList curIf in ifAllCond)
-{
-   try
-   {
-       IfStmt ifStmt = curIf.TryGetCSharpGraph<IfStmt>();
-       CxList cond = All.NewCxList();
-       cond.Add(ifStmt.Condition.NodeId, ifStmt.Condition);
-       conditions.Add(All.GetByAncs(cond));
-   }
-   catch (Exception e)
-   {
-       cxLog.WriteDebugMessage(e.Message);
-   }
-}
+CxList conditions = All.NewCxList(All.GetByAncs(ifAllCond.CxSelectDomProperty<IfStmt>(x => x.Condition)));

 //now get the variables, parameters and member accesses from all the elements in the conditions of ifs
-CxList relevant_objects = All.NewCxList();
 CxList allVariables = Find_UnknownReference();
 allVariables.Add(Find_Methods());
-relevant_objects.Add(allVariables, Find_Params(), Find_MemberAccess());
+CxList relevant_objects = All.NewCxList(allVariables, Find_Params(), Find_MemberAccess());

 conditions = conditions * relevant_objects;
```

Java / Java_Best_Coding_Practice / Uncontrolled_Recursion

Code changes

```
---
+++
@@ -30,8 +30,7 @@
 CxList iterationStmt = Find_IterationStmt();
 CxList TernaryExpr = Find_TernaryExpr();

-CxList conditionStatement = All.NewCxList();
-conditionStatement.Add(ForEachStmt, iterationStmt, TernaryExpr);
+CxList conditionStatement = All.NewCxList(ForEachStmt, iterationStmt, TernaryExpr);
 conditionStatement = conditionStatement.GetByAncs(methodOfMethod);
 conditionStatement.Add(IfStmt);
```

Java / Java_Best_Coding_Practice / Undocumented_API

Code changes

```
---
+++
@@ -1,7 +1,3 @@
-CxList customAttributes = Find_CustomAttribute();
-CxList objectCreations = Find_Object_Create();
-CxList methodDecls = Find_MethodDeclaration();
-
 // support for springfox (Swagger/OpenApi 1.2, 2.0 -- aka swagger-springmvc)
 // http://springfox.github.io/springfox/docs/current/

@@ -10,13 +6,13 @@
 // or @EnableSwaggerWebMvc are added to the config class
 // 2. a Docket object is created in a method in that class (with DocumentationType.SWAGGER/SWAGGER2 as parameter)

-List<string> springFoxAnnotationNames = new List<string>() {"EnableSwagger", "EnableSwagger2", "EnableSwagger2WebMvc"};
-CxList springFoxConfigAnnotations = customAttributes.FindByShortNames(springFoxAnnotationNames);
+string[] springFoxAnnotationNames = new string[] {"EnableSwagger", "EnableSwagger2", "EnableSwagger2WebMvc"};
+CxList springFoxConfigAnnotations = Find_CustomAttribute().FindByShortNames(springFoxAnnotationNames);
 CxList springFoxAnnotatedClasses = springFoxConfigAnnotations.GetAncOfType<ClassDecl>();
-CxList methodsOfAnnotatedClasses = methodDecls.GetByAncs(springFoxAnnotatedClasses);
+CxList methodsOfAnnotatedClasses = Find_MethodDeclaration().GetByAncs(springFoxAnnotatedClasses);

-CxList springFoxDocketObjects = objectCreations.FindByShortName("Docket").GetByAncs(methodsOfAnnotatedClasses);
-CxList springFoxDocumentationTypes = All.FindByMemberAccesses(new string[] {"DocumentationType.SWAGGER_12", "DocumentationType.SWAGGER_2", "DocumentationTye.SPRING_WEB"}, false);
+CxList springFoxDocketObjects = Find_Object_Create().FindByShortName("Docket").GetByAncs(methodsOfAnnotatedClasses);
+CxList springFoxDocumentationTypes = Find_Params().FindByMemberAccesses(new string[] {"DocumentationType.SWAGGER_12", "DocumentationType.SWAGGER_2", "DocumentationTye.SPRING_WEB"}, false);
 CxList relevantSpringFoxDocketObjects = springFoxDocketObjects.FindByParameters(springFoxDocumentationTypes);

 // If there are no such annotations and object creations, then search for other api documentation approaches
```

Java / Java_Best_Coding_Practice / Unused_Variable

Code changes

```
---
+++
@@ -3,29 +3,20 @@
 2. Variables that are initialized but never used.
 */

-CxList partialResults = All.NewCxList();
 /// Part 1 - Variables and function parameters that are never ever used
 CxList neverUsed = Unused_Variables_And_Functions_Params();

 /// Part 2 - Variables that are initialized but never used
 CxList onlyInitialized = Unused_Initialized_Variables();

-partialResults.Add(neverUsed, onlyInitialized);
-
-partialResults -= Find_Properties_Files();
```

```
-partialResults -= partialResults.FindAllReferences(All.GetByAncs(Find_Conditions()));
+CxList partialResults = All.NewCxList(neverUsed, onlyInitialized);

 //fmt_message_Key and fmt_bundle_file exists in FMT taglib of jsp files
 CxList fmt = partialResults.FindByFileName("*.MF");
 fmt.Add(partialResults.FindByShortNames(new string[] {"fmt_message_Key", "fmt_bundle_file*"}));
-partialResults -= fmt;
-
-// remove interface method params and variables
-partialResults -= partialResults.GetByAncs(Find_InterfaceDecl());

 CxList usedAssignRef = Find_Elimination_Variables();
 usedAssignRef = usedAssignRef.GetFirstNodesInPath();
-partialResults -= partialResults.FindAllReferences(usedAssignRef);

 // Remove lambda and @override methods parameters
 CxList lambdasAndOverrideMethods = Find_LambdaExpr();
@@ -34,28 +25,34 @@
 lambdasAndOverrideMethods.Add(overrideCustomAttr.GetAncOfType<MethodDecl>());

 CxList paramsToRemove = All.GetParameters(lambdasAndOverrideMethods);
-partialResults -= paramsToRemove;
-
-// Exclude JSF temp variables
-partialResults -= Find_JSF_Temp_Variables();
-
-//Exclude MyBatis temp variables
-partialResults -= Find_MyBatis_Temp_Variables();

 //Exclude Struts context variables
 CxList viewDeclarations = Find_ViewDecls();
 CxList paramDecls = Find_ParamDecl();
 CxList viewParameters = paramDecls.GetByAncs(viewDeclarations);
 CxList ctxParams = paramDecls.FindByShortName("ctx");
-partialResults -= ctxParams.GetParameters(viewDeclarations);
-//Exclude Struts generated resource variables
-partialResults -= partialResults.FindByShortName("appRes")
-   .FindByType("ApplicationResources").GetByAncs(viewDeclarations);
-//Exclude FF generated variables
-partialResults -= viewParameters;
-partialResults -= partialResults.FindByShortName("params").FindByType("Map");

-//Exclude the cases created by fmt_setBundle support
-partialResults -= partialResults.FindByType("ResourceBundle");
+CxList toRemove = All.NewCxList(
+   Find_Properties_Files(),
+   partialResults.FindAllReferences(All.GetByAncs(Find_Conditions())),
+   fmt,
+   // remove interface method params and variables
+   partialResults.GetByAncs(Find_InterfaceDecl()),
+   partialResults.FindAllReferences(usedAssignRef),
+   paramsToRemove,
+   // Exclude JSF temp variables
+   Find_JSF_Temp_Variables(),
+   //Exclude MyBatis temp variables
+   Find_MyBatis_Temp_Variables(),
+   ctxParams.GetParameters(viewDeclarations),
+   //Exclude Struts generated resource variables
+   partialResults.FindByShortName("appRes").FindByType("ApplicationResources").GetByAncs(viewDeclarations),
+   //Exclude FF generated variables
+   viewParameters,
+   partialResults.FindByShortName("params").FindByType("Map"),
+   //Exclude the cases created by fmt_setBundle support
+   partialResults.FindByType("ResourceBundle"));
+
+partialResults -= toRemove;

 result = partialResults;
```

Java / Java__Best__Coding__Practice / Use__of__Inner__Class__Containing__Sensitive__Data

Code changes

```
---
+++
@@ -1,6 +1,5 @@
 CxList classes = Find_Class_Decl();
-CxList appletClass = classes.InheritsFrom("Applet");
-appletClass.Add(classes.InheritsFrom("JApplet"));
+CxList appletClass = classes.InheritsFrom(new string [] {"Applet", "JApplet"});

 CxList innerClass = classes.GetByAncs(appletClass) - appletClass;
 CxList staticInnerClass = innerClass.FindByFieldAttributes(Modifiers.Static);
```

Java / Java__Best__Coding__Practice / Use__of__Obsolete__Functions

Code changes

```
---
+++
@@ -1,5 +1,5 @@
-result = All.NewCxList();
-result.Add(Find_CORBA_Deprecated_Methods(),
+result = All.NewCxList(
+           Find_CORBA_Deprecated_Methods(),
            Find_Java_Awt_Deprecated_Methods(),
            Find_Java_IO_Deprecated_Methods(),
            Find_Java_Lang_Deprecated_Methods(),
```

Java / Java_Best_Coding_Practice / Use_of_System_Output_Stream

Code changes

```
---
+++
@@ -1,5 +1,4 @@
 if(All.isWebApplication)
 {
-   CxList methods = Find_Methods();
-   result = methods.FindByMemberAccesses(new string [] {"JspWriter.print*", "out.print*", "err.print*"});
+   result = Find_Methods().FindByMemberAccesses(new string [] {"JspWriter.print*", "out.print*", "err.print*"});
 }
```

Java / Java_GWT / GWT_DOM_XSS

Code changes

```
---
+++
@@ -8,8 +8,7 @@
    // remove Reflected XSS outputs
    outputs -= outputs.GetByMethod(Find_MethodDecls().FindByShortName("onSuccess"));

-   CxList sanitize = All.NewCxList();
-   sanitize.Add(
+   CxList sanitize = All.NewCxList(
        All.FindByName("*encode*", false),
        Find_Methods().FindByShortName("toSafeHtml"));
```

Java / Java_GWT / GWT_Reflected_XSS

Code changes

```
---
+++
@@ -2,8 +2,7 @@

 if (gwtImports.Count > 0)
 {
-   CxList inputs = All.NewCxList();
-   inputs.Add(
+   CxList inputs = All.NewCxList(
        Find_GWT_Inputs(),
        Find_Interactive_Inputs());
```

Java / Java_GWT / JSON_Hijacking

Code changes

```
---
+++
@@ -1,9 +1,8 @@
 //DWR framework prevents javascript hijacking
 CxList dwrFramework = All.FindByName("*dwr.util*", true);

-CxList CleanAJAXFramework = All.NewCxList();
 // we'll add other frameworks that take care of javascript hijacking to this list
-CleanAJAXFramework.Add(dwrFramework);
+CxList CleanAJAXFramework = All.NewCxList(dwrFramework);

 if (CleanAJAXFramework.Count == 0)
 {
@@ -13,8 +12,7 @@

    if (json.Count > 0)
    {
-       CxList names = All.NewCxList();
-       names.Add(All.FindByNames(new string [] {"*select*", "*exec*"}, false));
+       CxList names = All.NewCxList(All.FindByNames(new string [] {"*select*", "*exec*"}, false));
        CxList db = Find_DB_Out().DataInfluencedBy(names);
        result = json.DataInfluencedBy(db).ReduceFlow(CxList.ReduceFlowType.ReduceSmallFlow);
    }
```

Java / Java_High_Risk / JSF_Local_File_Inclusion

Code changes

```
---
+++
@@ -1,6 +1,5 @@
 CxList inputs = All.NewCxList(Find_Inputs(), Find_Queue_Inputs());
-CxList allTypes = All.NewCxList();
-allTypes.Add(Find_TypeRef(), Find_Declarators(), Find_ObjectCreations(), Find_UnknownReference());
+CxList allTypes = All.NewCxList(Find_TypeRef(), Find_Declarators(), Find_ObjectCreations(), Find_UnknownReference());

 CxList dynamicTemplates = allTypes.FindByType("CxDynamicTemplateImport");
```

Java / Java_High_Risk / Second_Order_SQL_Injection

Code changes

```
---
+++
@@ -19,11 +19,9 @@
 CxList dbWithParams = dbIn.FindByParameters(dbParams);
 CxList dbWithNoParams = dbIn - dbWithParams;

-CxList endDB = All.NewCxList();
-endDB.Add(dbParams, dbWithNoParams);
+CxList endDB = All.NewCxList(dbParams, dbWithNoParams);

-CxList dbOutRead = All.NewCxList();
-dbOutRead.Add(dbOut, read);
+CxList dbOutRead = All.NewCxList(dbOut, read);
```

```
 CxList unknRefs = Find_UnknownReference();

 CxList xml = All.FindByFileName("*.xml");
```

Java / Java_High_Risk / Stored_XSS

Code changes

```
---
+++
@@ -5,15 +5,14 @@
    new string [] {"System.getProperty","System.getProperties"});
 readNonDB -= listSystemGetPropertiesInInputs;

-CxList read = All.NewCxList();
-read.Add(readNonDB);
+CxList read = All.NewCxList(readNonDB);
 // Remove Properties as they are considered potential inputs and are handled by the Potential_Stored_XSS query
 read -= read.FindByMemberAccess("Properties.getProperty");

 CxList getRequestSessionMethods = Find_GET_Request_Session_Methods();

-CxList inputs = Find_DB_Out();
-inputs.Add(
+CxList inputs = All.NewCxList(
+   Find_DB_Out(),
    read,
    Find_Vulnerable_Nio_Files_Methods(),
    Find_Vulnerable_Io_File_Methods(),
```

Java / Java_Low_Visibility / Authorization_Bypass_Through_User_Controlled_SQL_PrimaryKey

Code changes

```
---
+++
@@ -1,14 +1,10 @@
-// database
-CxList db = Find_DB_In();
-// strings
-CxList strings = Find_Strings();
 // strings that end with "id" (there might be also "pid" and others, but then I'm starting with many
 // potential false positives
-CxList id = strings.FindByShortName("*id\"");
+CxList id = Find_Strings().FindByShortName("*id\"");
 // Interative inputs, that are influenced by this id (usually getParameter or alike)
 CxList input = Find_Interactive_Inputs();
 input = input.DataInfluencedBy(id);



 /// DB influenced by potentially problematic input
-result = db.DataInfluencedBy(input).DataInfluencedBy(id);
```

```
+result = Find_DB_In().DataInfluencedBy(input).DataInfluencedBy(id);
```

Java / Java_Low_Visibility / Channel_Accessible_by_NonEndpoint

Code changes

```
---
+++
@@ -1,3 +1,5 @@
+CxList methods = Find_Methods();
+
 // Find all outputs that have "print*" or "write*"
 CxList outputsPrintWrites = Find_Outputs();
 CxList outputs = outputsPrintWrites.FindByShortNames(new string [] {"print*", "write*"});
@@ -13,11 +15,11 @@
 */

 // When the output is Socket, it is not secured (not SSLSocket)
-CxList Socket = All.FindByMemberAccess("Socket.getOut*").GetTargetOfMembers();
+CxList Socket = methods.FindByMemberAccess("Socket.getOut*").GetTargetOfMembers();
 //Socket.Add(All.FindByMemberAccess("SocketChannel.open"));

 //Secure
-CxList wrapSSL = All.FindByMemberAccess("SSLEngine.wrap");
+CxList wrapSSL = methods.FindByMemberAccess("SSLEngine.wrap");
 //Parameters that are secure
 CxList wrap_param = All.FindAllReferences(All.GetParameters(wrapSSL, 1));//Get output from wrap(passed by reference)
 //Outputs that use secure parameters
```

Java / Java_Low_Visibility / Cleansing_Canonicalization_and_Comparison_Errors

Code changes

```
---
+++
@@ -1,9 +1,11 @@
-CxList inputs = All.FindByMemberAccesses(new string[]{
+CxList methods = Find_Methods();
+
+CxList inputs = methods.FindByMemberAccesses(new string[]{
    "*HttpServletRequest.getRequestURI",
    "*HttpServletRequest.getRequestURL",
    "*HttpServletRequest.getServletPath"});

-CxList sanitize = All.FindByMemberAccesses(new string[]{
+CxList sanitize = methods.FindByMemberAccesses(new string[]{
    "URLDecoder.decode",
    "Encoder.decodeForURL"});

@@ -18,7 +20,8 @@

 CxList nullFathers = Find_NullLiteral().GetFathers() * binaryExpr;
```

```
-binaryExpr -= emptyStringBinaryExpr;
-binaryExpr -= nullFathers;
+CxList toRemove = All.NewCxList(emptyStringBinaryExpr, nullFathers);
+
+binaryExpr -= toRemove;

 result = binaryExpr.InfluencedByAndNotSanitized(inputs, sanitize);
```

Java / Java_Low_Visibility / Collapse_of_Data_into_Unsafe_Value

Code changes

```
---
+++
@@ -1,9 +1,8 @@
 CxList inputs = Find_Interactive_Inputs();
 CxList outputs = Find_XSS_Outputs();
-CxList methods = Find_Methods();

 // Find all Replace
-CxList replace = methods.FindByShortName("replace*");
+CxList replace = Find_Methods().FindByShortName("replace*");

 // Look at replace that is affected by inputs, and affecting outputs (potential XSS),
 // but does not pass through a sanitizer
```

Java / Java_Low_Visibility / Cookie_Overly_Broad_Path

Code changes

```
---
+++
@@ -14,39 +14,13 @@
 CxList binary = Find_BinaryExpr();
 // Reduce the binary list for the sake of performance
 binary = binary.InfluencingOn(setPath);
-foreach(CxList l in binary)
-{
-   try
-   {
-       BinaryExpr b = l.TryGetCSharpGraph<BinaryExpr>();
-       if(b != null && b.Operator.ToString() == "Add")
-       {
-           concat.Add(l);
-       }
-   }
-   catch(Exception ex)
-   {
-       cxLog.WriteDebugMessage(ex);
-   }
-}
```

```
+concat.Add(binary.FilterByDomProperty<BinaryExpr>(x => x != null && x.Operator.ToString() == "Add"));


 // Find the '+=' operators
 CxList assignments = Find_AssignExpr();
 // Reduce the assignments list for the sake of performance
 assignments = assignments.InfluencingOn(setPath);
-foreach(CxList assignment in assignments)
-{
-   try
-   {
-       AssignExpr graph = assignment.TryGetCSharpGraph<AssignExpr>();
-       if(graph != null && graph.Operator == AssignOperator.AdditionAssign)
-           concat.Add(assignment);
-   }
-   catch(Exception ex)
-   {
-       cxLog.WriteDebugMessage(ex);
-   }
-}
+concat.Add(assignments.FilterByDomProperty<AssignExpr>(x => x != null && x.Operator == AssignOperator.AdditionAssign));


 // Find places where the concat is in the setPath call (e.g. cookie.setPath(str1 + "/");)
 CxList concatInSetPath = concat.GetByAncs(setPath);
```

Java / Java_Low_Visibility / Creation_of_Temp_File_in_Dir_with_Incorrect_Permissions

Code changes

```
---
+++
@@ -1,8 +1,10 @@
-CxList createFile = All.FindByMemberAccesses(new string [] {"File.createNewFile", "File.createTempFile"});
+CxList methods = Find_Methods();
+
+CxList createFile = methods.FindByMemberAccesses(new string [] {"File.createNewFile", "File.createTempFile"});


 CxList newFileObject = All.FindByShortName("File").GetAncOfType<ObjectCreateExpr>();


-CxList permissions = All.FindByMemberAccesses(new string [] {"File.setExecutable", "File.setReadable", "File.setWritable"});
+CxList permissions = methods.FindByMemberAccesses(new string [] {"File.setExecutable", "File.setReadable", "File.setWritable"});


 CxList insecureCreate = createFile - createFile.DataInfluencedBy(permissions);
 CxList createInfluenced = newFileObject.DataInfluencingOn(insecureCreate);
```

Java / Java_Low_Visibility / Creation_of_Temp_File_With_Insecure_Permissions

Code changes

```
---
+++
@@ -1,8 +1,11 @@
-CxList createNewFile = All.FindByMemberAccess("File.createTempFile");
```

```
+CxList methods = Find_Methods();


-CxList writeToFile = Find_Methods().FindByShortName("write*");

+CxList createNewFile = methods.FindByMemberAccess("File.createTempFile");


-CxList permissions = All.FindByMemberAccesses(new string [] {"File.setExecutable", "File.setReadable", "File.setWritable"});

+CxList writeToFile = methods.FindByShortName("write*");

+

+CxList permissions = methods.FindByMemberAccesses(new string [] {"File.setExecutable",

+   "File.setReadable", "File.setWritable"});


 CxList insecureWrite = writeToFile - writeToFile.DataInfluencedBy(permissions);

 result = createNewFile.DataInfluencingOn(insecureWrite);
```

Java / Java__Low__Visibility / Divide__By__Zero

Code changes

```
---

+++

@@ -35,8 +35,7 @@


 CxList cond = unknown * All.GetByAncs(Find_Conditions());


-CxList sanitize = All.NewCxList();

-sanitize.Add(bin, indexer, methods);

+CxList sanitize = All.NewCxList(bin, indexer, methods);


 CxList divBin = bin.FilterByDomProperty<BinaryExpr>(x => x.Operator == BinaryOperator.Divide ||

    x.Operator == BinaryOperator.Modulus);

@@ -105,7 +104,7 @@


 // Division by input or Random

 CxList inputs = Find_Inputs(); // input

-inputs.Add(All.FindByMemberAccesses(new string [] {"Random.next*", // Random (1)

+inputs.Add(methods.FindByMemberAccesses(new string [] {"Random.next*", // Random (1)

                                                     "Math.random"}),

        Get_ESAPI().FindByMemberAccess("Randomizer.*")); // ESAPI
```

Java / Java__Low__Visibility / ESAPI__Same__Password__Repeats__Twice

Code changes

```
---

+++

@@ -5,13 +5,14 @@


 */


-CxList createUser = All.FindByMemberAccess("authenticator.createUser");

+CxList createUser = Find_Methods().FindByMemberAccess("authenticator.createUser");
```

```
 CxList inputs = Find_Interactive_Inputs();
+CxList unkRefs = Find_UnknownReference();

 // Find password parameters in createUser
-CxList param1 = All.GetParameters(createUser, 1).FindByType<UnknownReference>();
-CxList param2 = All.GetParameters(createUser, 2).FindByType<UnknownReference>();
+CxList param1 = unkRefs.GetParameters(createUser, 1);
+CxList param2 = unkRefs.GetParameters(createUser, 2);

 // Leave only parameters that are influenced by input
 param1 = param1 * param1.DataInfluencedBy(inputs);
```

Java / Java_Low_Visibility / Exposure_of_System_Data

Code changes

```
---
+++
@@ -2,8 +2,7 @@

 CxList getFromSystem = Find_Methods().FindByMemberAccess("System.getenv");

-CxList inputs = All.NewCxList();
-inputs.Add(getFromSystem);
+CxList inputs = All.NewCxList(getFromSystem);

 CxList interactiveOutputs = Find_Interactive_Outputs();
```

Java / Java_Low_Visibility / File_Permissions_World_Readable

Code changes

```
---
+++
@@ -1,9 +1,11 @@
 //File permission World readable

-CxList fileReadable = All.FindByMemberAccess("File.setReadable");
+CxList fileReadable = Find_Methods().FindByMemberAccess("File.setReadable");
+
+CxList booleanLits = Find_BooleanLiteral();

-CxList firstPrm = All.GetParameters(fileReadable, 0).FindByType<BooleanLiteral>(); // true: File Readable
-CxList secondPrm=All.GetParameters(fileReadable, 1).FindByType<BooleanLiteral>();// true: Readable for the owner
+CxList firstPrm = booleanLits.GetParameters(fileReadable, 0); // true: File Readable
+CxList secondPrm = booleanLits.GetParameters(fileReadable, 1);// true: Readable for the owner
 CxList analyzePermissions = All.NewCxList();
 if (firstPrm.FindByAbstractValue(abstractValue => abstractValue is TrueAbstractValue).Count > 0)
 {
```

Java / Java_Low_Visibility / Heap_Inspection

Code changes

```
---
+++
@@ -6,12 +6,13 @@

 CxList objectCreations = Find_ObjectCreations();
 CxList passwords = Find_Passwords_Unsafe();
+CxList methods = Find_Methods();

 // remove passwords used as flags (Booleans)
-passwords -= passwords.FindByTypes(new string[]{"bool", "boolean", "Cipher"}, false);

-// remove passwords files and paths
-passwords -= passwords.FindByTypes(new string[]{"File*", "Path"}, false);
+passwords -= passwords.FindByTypes(new string[]{"bool", "boolean", "Cipher",
+   // remove passwords files and paths
+   "File*", "Path"}, false);

 //remove types that are defined in scanned code
 //the result of heap inspection should be inside of the class
@@ -19,16 +20,8 @@
 System.Text.RegularExpressions.Regex re = new System.Text.RegularExpressions.Regex(pattern);
 CxList typesDefined = Find_ClassDecl();
 CxList toRemove = All.NewCxList();
-foreach(CxList types in typesDefined){
-   try{
-       CSharpGraph typeGraph = types.GetFirstGraph();
-       string path = typeGraph.LinePragma.FileName;
-       if(re.IsMatch(path)){
-           toRemove.Add(types);
-       }
-   }
-   catch(Exception){}
-}
+toRemove.Add(typesDefined.FilterByDomProperty<ClassDecl>(x => re.IsMatch(x.LinePragma.FileName)));
+
 typesDefined -= toRemove;
 CxList typeRefs = Find_TypeRef();
 CxList allRefs = typeRefs.FindAllReferences(typesDefined);
@@ -40,7 +33,7 @@
 passwords -= passwords.GetByAncs(passwordVariables * variables);

 //2) define sanitizers = encryption
-CxList sanitizeMethods = All.FindByMemberAccesses(new string [] {"KeyStore.setKeyEntry", "KeyStore.SetCertificate"});
+CxList sanitizeMethods = methods.FindByMemberAccesses(new string [] {"KeyStore.setKeyEntry", "KeyStore.SetCertificate"});
 sanitizeMethods = objectCreations.FindByShortName("SealedObject");
 CxList sanitize = All.FindByFathers(sanitizeMethods);
 sanitize.Add(Find_Encrypt(), Find_HashSanitize());
@@ -93,7 +86,7 @@

 result -= result.FindDefinition(safePasswords);

-CxList fillSanitizer = All.FindByMemberAccess("Arrays.fill");
+CxList fillSanitizer = methods.FindByMemberAccess("Arrays.fill");
 CxList fillParameters = All.GetParameters(fillSanitizer, 0);

 result -= result.FindDefinition(fillParameters);
```

Java / Java_Low_Visibility / Improper_Resource_Access_Authorization

Code changes

```
---
+++
@@ -10,8 +10,7 @@

 // Find database and file accesses
 CxList db = Find_DB();
-CxList dataAccess = All.NewCxList();
-dataAccess.Add(db, Find_IO(), fileReads, Find_FileSystem_Write());
+CxList dataAccess = All.NewCxList(db, Find_IO(), fileReads, Find_FileSystem_Write());

 // Only consider methods
 dataAccess = dataAccess.FindByType<MethodInvokeExpr>();
@@ -31,10 +30,7 @@
 testAndOtherItems.Add(mainMethods, methodDecls.FindByShortName("CxStaticBlock1"));
 dataAccess -= dataAccess.GetByAncs(testAndOtherItems);

-CxList suspectConditionParams = All.NewCxList();
-
-suspectConditionParams.Add(unkRefs, memberAccesses, methods);
-
+CxList suspectConditionParams = All.NewCxList(unkRefs, memberAccesses, methods);

 CxList toRemove = All.NewCxList();

@@ -59,8 +55,8 @@
    };
 toRemove.Add(dataAccess.FindByMemberAccesses(toRemoveNames), dataAccess.FindByShortName("getenv"));

-CxList searchSpace = All.NewCxList();
-searchSpace.Add(unkRefs, declarators);
+CxList searchSpace = All.NewCxList(unkRefs, declarators);
+
 toRemove.Add(searchSpace.InfluencedBy(db));

 dataAccess -= toRemove;
```

```
@@ -185,7 +181,7 @@

 // Find conditions that make use of *auth* and *admin* words //- heuristics
 CxList conditions = suspectConditionParams.GetByAncs(Find_Conditions());
-List<string> possibleStr = new List<string>(){
+string[] possibleStr = new string[]{
        "*admin*", "*allow",
        "*allowed", "*allows", "*deny", "*denies", "*denied",
        "*authoriz*", "*permission*"
@@ -202,8 +198,7 @@
 read.Add(fileReads);
 CxList unsafeReads = dataAccessUnsafe * read;
 CxList unsafeWriteMethods = dataAccessUnsafe - unsafeReads;
-CxList unsafeParams = All.NewCxList();
-unsafeParams.Add(unkRefs, methods, memberAccesses, paramss);
+CxList unsafeParams = All.NewCxList(unkRefs, methods, memberAccesses, paramss);
 CxList flowInputs = inputs.InfluencingOn(unsafeParams.GetParameters(unsafeWriteMethods));

 result = unsafeReads;
```

Java / Java_Low_Visibility / Improper_Resource_Locking

Code changes

```
---
+++
@@ -1,3 +1,3 @@
-CxList tryLock = All.FindByMemberAccess("ReentrantLock.tryLock");
+CxList tryLock = Find_Methods().FindByMemberAccess("ReentrantLock.tryLock");

 result = tryLock - Find_Conditions();
```

Java / Java_Low_Visibility / Improper_Resource_Shutdown_or_Release

Code changes

```
---
+++
@@ -106,21 +106,8 @@
 CxList TryEnds = allResourcesInProj.GetLastNodesInPath();

 CxList TryBlocks = All.NewCxList();
-foreach(CxList tryCatch in trys)
-{
-   try
-   {
-       TryCatchFinallyStmt tryGraph = tryCatch.TryGetCSharpGraph<TryCatchFinallyStmt>();
-       if(tryGraph.Try != null)
-       {
-           TryBlocks.Add(tryGraph.Try.NodeId, tryGraph.Try);
-       }
-   }
-   catch(Exception ex)
-   {
-       cxLog.WriteDebugMessage(ex);
-   }
-}
+TryBlocks.Add(trys.CxSelectDomProperty<TryCatchFinallyStmt>(x => x.Try));
+
 CxList nodeObjectDeclarator = allResourcesInProj.GetFirstNodesInPath();
 nodeObjectDeclarator.Add(wrappingObjects);

@@ -213,11 +200,7 @@

 // Get all classes inherits from autocloseable
 CxList classNames = classDecls.InheritsFrom("AutoCloseable");
-foreach(CxList c in classNames)
-{
-   string clsName = c.GetName();
-   autocloseable.Add(clsName);
-}
+autocloseable.AddRange(classNames.CxSelectElementValues<ClassDecl, string>(x => x.ShortName));

 CxList paramOfBuffered = All.GetParameters(All.FindByShortNames(autocloseable));
 CxList closedByBuffered = tempResult.InfluencingOn(paramOfBuffered);
```

Java / Java_Low_Visibility / Information_Exposure_Through_an_Error_Message

Code changes

```
---
+++
@@ -12,8 +12,7 @@
 CxList excDecl = variableDecls.FindByFathers(ctch);
 CxList exp = exception.GetByAncs(excDecl);

-CxList methodAndMember = All.NewCxList();
-methodAndMember.Add(Find_Methods(), Find_MemberAccesses());
+CxList methodAndMember = All.NewCxList(Find_Methods(), Find_MemberAccesses());

 CxList exceptionsOutsideCatch = methodAndMember.FindByMemberAccess("Exception.*").GetTargetOfMembers();
 exceptionsOutsideCatch -= exceptionsOutsideCatch.FindByType<BaseRef>();
@@ -21,14 +20,13 @@

 //find the outputs that are influenced by an Exception as a cast
 //example: ((Exception)r.GetException()).printStackTrace();
-CxList sourcetype = All.NewCxList();
-sourcetype.Add(
+CxList sourcetype = All.NewCxList(
    Find_BaseRef(),
    Find_ClassDecl(),
    Find_UnknownReference(),
    Find_Object_Create(),
    Find_TypeRef(),
-   declarators);
+   declarators);

 CxList allExceptions = sourcetype.FindByTypes(new String[]{"*Exception","Throwable"});
 allExceptions = allExceptions.GetAncOfType<CastExpr>();
```

Java / Java_Low_Visibility / Information_Exposure_Through_Debug_Log

Code changes

```
---
+++
@@ -1,11 +1,12 @@
 CxList deadCode = Find_Dead_Code_Contents();
+CxList methods = Find_Methods();

-CxList getSession = All.FindByMemberAccesses(new string [] {"HttpServletRequest.getSession",
+CxList getSession = methods.FindByMemberAccesses(new string [] {"HttpServletRequest.getSession",
                                                             "*response.getSession",
                                                             "*Response.getSession"});

 CxList inputs = getSession.GetMembersOfTarget().FindByShortName("getId");
-inputs.Add(All.FindByMemberAccess("HttpSession.getId"));
+inputs.Add(methods.FindByMemberAccess("HttpSession.getId"));

 CxList outputs = Find_Log_Outputs();
```

Java / Java_Low_Visibility / Information_Exposure_Through_Query_String

Code changes

```
---
+++
@@ -1,5 +1,4 @@
-CxList sensitiveInformation = All.NewCxList();
-sensitiveInformation.Add(Find_Personal_Info(), Find_Password_Info());
+CxList sensitiveInformation = All.NewCxList(Find_Personal_Info(), Find_Password_Info());
 CxList passwordRelatedNodes = Password_Privacy_Violation_List();
 passwordRelatedNodes.Add(Find_Password_Strings());
 CxList methods = Find_Methods();
```

Java / Java_Low_Visibility / Information_Exposure_Through_Server_Log

Code changes

```
---
+++
@@ -1,15 +1,14 @@
 CxList methods = Find_Methods();
-CxList deadCode = Find_Dead_Code_Contents();
```

```
-CxList getSession = All.FindByMemberAccess("HttpServletRequest.getSession");
+CxList getSession = methods.FindByMemberAccess("HttpServletRequest.getSession");
 getSession.Add(All.FindByName("*response.getSession", false));

 CxList inputs = getSession.GetMembersOfTarget().FindByShortName("getId");
-inputs.Add(All.FindByMemberAccess("HttpSession.getId"));
+inputs.Add(methods.FindByMemberAccess("HttpSession.getId"));

 CxList outputs = methods.FindByName("log", false);

 CxList sanitize = Find_Integers();
-sanitize.Add(deadCode);
+sanitize.Add(Find_Dead_Code_Contents());

 result = outputs.InfluencedByAndNotSanitized(inputs, sanitize);
```

Java / Java_Low_Visibility / Information_Leak_Through_Shell_Error_Message

Code changes

```
---
+++
@@ -1,14 +1,14 @@
-CxList deadCode = Find_Dead_Code_Contents();
+CxList methods = Find_Methods();

-CxList getSession = All.FindByMemberAccess("HttpServletRequest.getSession");
+CxList getSession = methods.FindByMemberAccess("HttpServletRequest.getSession");
 getSession.Add(All.FindByName("*response.getSession", false));

 CxList inputs = getSession.GetMembersOfTarget().FindByShortName("getId");
-inputs.Add(All.FindByMemberAccess("HttpSession.getId"));
+inputs.Add(methods.FindByMemberAccess("HttpSession.getId"));

 CxList outputs = Find_Console_Outputs();

 CxList sanitize = Find_Integers();
-sanitize.Add(deadCode);
+sanitize.Add(Find_Dead_Code_Contents());

 result = outputs.InfluencedByAndNotSanitized(inputs, sanitize);
```

Java / Java_Low_Visibility / Insufficiently_Protected_Credentials

Code changes

```
---
+++
@@ -1,8 +1,8 @@
-CxList psw  = Find_Passwords();
+CxList psw = Find_Passwords();
 psw -= Find_Methods();
 CxList DB = Find_DB_Out();
 CxList sanitize = Find_General_Sanitize();
-sanitize += Find_Decrypt();
+sanitize.Add(Find_Decrypt());

-result = DB.InfluencingOnAndNotSanitized(psw, sanitize);
+result = DB.InfluencingOnAndNotSanitized(psw, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Low_Visibility / Insufficient_Session_Expiration

Code changes

```
---
+++
@@ -3,7 +3,7 @@
 1. Session expiration in web.xml
 2. Session expiration in a Java (or jsp) file
 */
-
+CxList methods = Find_Methods();
 /// Case 1- Session expiration in web.xml

 //// Find in web.xml the session-timeout, if exists
@@ -19,11 +19,11 @@

 /// Case 2 - Session expiration in a Java (or jsp) file

-CxList getSession = All.FindByMemberAccess("HttpServletRequest.getSession");
+CxList getSession = methods.FindByMemberAccess("HttpServletRequest.getSession");
 getSession.Add(All.FindByName("*request.getSession", false));

 // Find all setMaxInactiveInterval in a session
-CxList maxInactiveInterval = All.FindByMemberAccess("HttpSession.setMaxInactiveInterval");
+CxList maxInactiveInterval = methods.FindByMemberAccess("HttpSession.setMaxInactiveInterval");
 maxInactiveInterval.Add(getSession.GetMembersOfTarget().FindByShortName("setMaxInactiveInterval"));

 maxInactiveInterval = All.GetParameters(maxInactiveInterval);
```

Java / Java_Low_Visibility / Integer_Overflow

Code changes

Java / Java_Low_Visibility / Integer_Underflow

Code changes

Java / Java_Low_Visibility / JWT_Excessive_Expiration_Time

Code changes

```
---
+++
@@ -27,7 +27,6 @@
 CxList excessiveValues = possibleParams.GetByAncs(dateParams)
    .FindByAbstractValue(absValue => acceptableRange.Contains(absValue)).GetFathers() * dateParams;

-CxList expParamExcVal = All.NewCxList();
-expParamExcVal.Add(expirationParams, excessiveValues);
+CxList expParamExcVal = All.NewCxList(expirationParams, excessiveValues);
 result = generators.DataInfluencedBy(expParamExcVal);
 result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Low_Visibility / JWT_Use_Of_None_Algorithm

Code changes

```
---
+++
@@ -1,7 +1,7 @@
 CxList methods = Find_Methods();
 CxList jwtsBuildRef = methods.FindByMemberAccess("Jwts.builder");
-CxList sanitizedMethod = All.FindByMemberAccess("JwtBuilder.*")
-   .FindByShortName("signWith").GetTargetOfMembers();
+
+CxList sanitizedMethod = methods.FindByMemberAccess("JwtBuilder.signWith").GetTargetOfMembers();
 sanitizedMethod.Add(methods.FindByShortName("signWith"));

 CxList lastNodeFlowJwts = methods.FindByShortName("compact");
```

Java / Java_Low_Visibility / Logic_Time_Bomb

Code changes

```
---
+++
@@ -5,7 +5,7 @@

 CxList conditions = Find_Conditions();

-CxList time_cond = All.FindByMemberAccesses(new string [] {"Calendar.after", "Duration.isZero", "Duration.isNegative",
+CxList time_cond = methods.FindByMemberAccesses(new string [] {"Calendar.after", "Duration.isZero", "Duration.isNegative",
                                                              "Instant.isAfter", "Instant.isZero", "LocalDate.isAfter",
                                                              "LocalDate.isBefore", "LocalDate.isEqual",
                                                              "LocalDate.isLeapYear", "LocalDateTime.isAfter",
```

Java / Java_Low_Visibility / Object_Hijack

Code changes

```
---
+++
@@ -1,5 +1,4 @@
 CxList cloneable = All.InheritsFrom("Cloneable");
-CxList methodDecl = Find_MethodDecls();
```

```
-CxList clone = methodDecl.GetByAncs(cloneable).FindByShortName("clone");
+CxList clone = Find_MethodDecls().GetByAncs(cloneable).FindByShortName("clone");

 result = clone - clone.FindByFieldAttributes(Modifiers.Sealed);
```

Java / Java_Low_Visibility / Portability_Flaw_Locale_Dependent_Comparison

Code changes

```
---
+++
@@ -2,7 +2,7 @@
 CxList methods = Find_Methods();

 //Find Locale senstive funtions
-CxList localeSensitiveMethods = methods.FindByShortNames(new List<string>{
+CxList localeSensitiveMethods = methods.FindByShortNames(new string[]{
        "toLowerCase",
        "toUpperCase"});

@@ -17,7 +17,7 @@
 localeSensitiveMethods -= sanitizedMethods;

 //Find String search or comparison methods
-CxList targetMethods = methods.FindByShortNames(new List<string>{
+CxList targetMethods = methods.FindByShortNames(new string[]{
        "compareTo*",
        "contains",
        "contentEquals",
@@ -30,9 +30,9 @@
        "replace"}, false);

 //Adds all methods from the class Strings - Java and Google Guava API
-targetMethods.Add(All.FindByMemberAccess("Strings", "*"));
+targetMethods.Add(methods.FindByMemberAccess("Strings", "*"));
 //Adds all methods from the class StringUtils - Apache Commons Lang3 API
-targetMethods.Add(All.FindByMemberAccess("StringUtils", "*"));
+targetMethods.Add(methods.FindByMemberAccess("StringUtils", "*"));

 result = targetMethods.DataInfluencedBy(localeSensitiveMethods);
 //
```

Java / Java_Low_Visibility / Potential_ReDoS

Code changes

```
---
+++
@@ -3,7 +3,6 @@

 CxList evilString = Find_Evil_Strings();
```

```
-CxList toRemove = All.NewCxList();
-toRemove.Add(filter, evilString.DataInfluencingOn(filter));
+CxList toRemove = All.NewCxList(filter, evilString.DataInfluencingOn(filter));

 result = evilString - toRemove;
```

Java / Java_Low_Visibility / Potential_ReDoS_In_Static_Field

Code changes

```
---
+++
@@ -5,7 +5,7 @@
 CxList sanitize = Find_Integers();

 // Find all regex commands
-CxList regex = All.FindByMemberAccess("Pattern.compile");
+CxList regex = Find_Methods().FindByMemberAccess("Pattern.compile");

 // Add static regexes (these do not influence their references, so needed here)
 CxList staticFields = All.FindByFieldAttributes(Modifiers.Static);
```

Java / Java_Low_Visibility / Race_Condition

Code changes

```
---
+++
@@ -5,8 +5,7 @@

 // Find all classes that are HTTP Servlets and Singleton
 CxList servletClasses = All.NewCxList();
-CxList singleLevel = classes.InheritsFrom("HttpServlet");
-singleLevel.Add(classes.InheritsFrom("HttpJspBase"));
+CxList singleLevel = classes.InheritsFrom(new string []{"HttpServlet", "HttpJspBase"});

 int levelCount = singleLevel.Count;
 int counter = 0;
@@ -47,7 +46,7 @@
    CxList fields = allClassFields - subClassesFields;
    // Remove variables assigned in init, run and destroy methods
    CxList classMethods = classChildren.FindByType<MethodDecl>();
-   CxList runInitMethods = classMethods.FindByShortNames(new List<string> {"init", "run", "destroy"}, false);
+   CxList runInitMethods = classMethods.FindByShortNames(new string[] {"init", "run", "destroy"}, false);
    CxList fieldAssignment = All.FindAllReferences(fields).FindByAssignmentSide(CxList.AssignmentSide.Left);
    CxList cleanFields = fieldAssignment.GetByAncs(runInitMethods);
    List<string> cleanFieldNames = new List<string>();
@@ -74,8 +73,7 @@

 //unknown referenses in constructors
 CxList allCostructorVars = allUnkownRefs.GetByAncs(Find_ConstructorDecl());
-CxList allUnkRefIndex = All.NewCxList();
```

```
-allUnkRefIndex.Add(allUnkownRefs, allIndexers);
+CxList allUnkRefIndex = All.NewCxList(allUnkownRefs, allIndexers);
 CxList allRefs = allUnkRefIndex - allUnkownRefs.GetByAncs(allIndexers);
 allRefs -= allCostructorVars;
```

Java / Java_Low_Visibility / Reliance_on_Cookies_in_a_Decision

Code changes

```
---
+++
@@ -1,5 +1,5 @@
 CxList cookies =
-   All.FindByMemberAccess("request.getCookies");
+   Find_Methods().FindByMemberAccess("request.getCookies");
 cookies.Add(Find_CookieValue_Annotation());

 CxList cond = Find_Conditions();
```

Java / Java_Low_Visibility / Reliance_on_DNS_Lookups_in_a_Decision

Code changes

```
---
+++
@@ -1,7 +1,7 @@
-CxList cond = Find_Conditions();
+CxList methods = Find_Methods();

-CxList ip = All.FindByMemberAccess("request.getRemoteAddr");
+CxList ip = methods.FindByMemberAccess("request.getRemoteAddr");

-CxList inetAddress = All.FindByMemberAccesses(new string [] {"InetAddress.getByName", "InetAddress.getByAddress"});
+CxList inetAddress = methods.FindByMemberAccesses(new string [] {"InetAddress.getByName", "InetAddress.getByAddress"});

-result = cond.DataInfluencedBy(inetAddress).DataInfluencedBy(ip);
+result = Find_Conditions().DataInfluencedBy(inetAddress).DataInfluencedBy(ip);
```

Java / Java_Low_Visibility / Sensitive_Cookie_in_HTTPS_Session_Without_Secure_Attribute

Code changes

```
---
+++
@@ -1,5 +1,7 @@
+CxList methods = Find_Methods();
+
 // Find the setSevcured(true)
-CxList setSecure = All.FindByMemberAccess("Cookie.setSecure");
+CxList setSecure = methods.FindByMemberAccess("Cookie.setSecure");
 CxList trues = All.FindByShortName("true");
 CxList secured = trues.GetParameters(setSecure);
```

```
@@ -10,7 +12,7 @@
    webFiles.FindByName("WEB_APP.SESSION_CONFIG.COOKIE_CONFIG.SECURE.TEXT").GetAssigner().FindByShortName("true").Count == 0)
 {
    // Find the added cookies
-   CxList addCookie = All.FindByMemberAccess("response.addCookie");
+   CxList addCookie = methods.FindByMemberAccess("response.addCookie");
    addCookie.Add(All.FindByNames("*response.addCookie","*Response.addCookie"));

    CxList cookies = All.GetParameters(addCookie).FindByTypes("*.Cookie","Cookie");
```

Java / Java_Low_Visibility / Serializable_Class_Containing_Sensitive_Data

Code changes

```
---
+++
@@ -1,6 +1,5 @@
 // Find Sensitive data field (non boolean)
-CxList sensitiveData = All.NewCxList();
-sensitiveData.Add(Find_Personal_Info(), Find_Password_Info());
+CxList sensitiveData = All.NewCxList(Find_Personal_Info(), Find_Password_Info());
 sensitiveData = sensitiveData * Find_Field_Decl();
 CxList removePart = sensitiveData.FindByTypes(new string[] {"boolean","bool"});
 sensitiveData -= removePart;
```

Java / Java_Low_Visibility / TOCTOU

Code changes

```
---
+++
@@ -15,16 +15,16 @@
 //     FileWriter fw = new FileWriter(f);
 //     fw.close();
 //
-
+CxList methods = Find_Methods();

 //Look for all f.canWrite() methods
-CxList canWriteMethods = All.FindByMemberAccess("File.canWrite");
+CxList canWriteMethods = methods.FindByMemberAccess("File.canWrite");

 // Choose all if statement that includes canWrite condition
 CxList ifStmt = canWriteMethods.GetAncOfType<IfStmt>();

 // choose all sleep methods
-CxList sleep = Find_Methods().FindByShortName("sleep");
+CxList sleep = methods.FindByShortName("sleep");
 sleep.Add(Find_Read_NonDB());

 // choose id statements that includes sleep and .canWrite
```

Java / Java__Low__Visibility / Unsynchronized__Access_To__Shared__Data

Code changes

```
---
+++
@@ -1,11 +1,12 @@
+CxList methods = Find_Methods();
+
 CxList logs = All.FindByTypes(new string [] {"Log", "Logger"});
 // Remove the ThreadLocal because it's a thread safe object
 CxList localThread = All.FindByType("ThreadLocal");
 // Remove Location.getLocation static field because it is never changed after its initialization
-CxList locations = All.FindByMemberAccess("Location.getLocation").GetAncOfType<FieldDecl>();
+CxList locations = methods.FindByMemberAccess("Location.getLocation").GetAncOfType<FieldDecl>();

-CxList toRemove = All.NewCxList();
-toRemove.Add(logs, localThread, locations);
+CxList toRemove = All.NewCxList(logs, localThread, locations);

 // Remove false sinks such asTypeRef and GenericTypeRef
 CxList noLogs = All - logs;
@@ -17,13 +18,13 @@
 statics -= unwanted;

 CxList staticsRefs = noLogs.FindAllReferences(statics);
-staticsRefs -= Find_Methods();
+staticsRefs -= methods;

 CxList staticDecl = staticsRefs.FindByType<Declarator>();
 staticsRefs -= staticDecl;

 CxList inputs = Find_Interactive_Inputs();
-inputs.Add(All.FindByMemberAccess("ServerRequest.getAttribute"));
+inputs.Add(methods.FindByMemberAccess("ServerRequest.getAttribute"));

 CxList threadSafetyIssue = inputs.InfluencingOnAndNotSanitized(staticsRefs, staticDecl);
```

Java / Java__Low__Visibility / Use_of__Broken__or__Risky__Cryptographic__Algorithm

Code changes

```
---
+++
@@ -7,21 +7,18 @@
 CxList toRemove = strings.FindByShortNames(new string [] {"*DESEDE*", "*TripleDES*"}, false);
 des -= toRemove;

-CxList weakTypes = All.NewCxList();
-weakTypes.Add(
+CxList weakTypes = All.NewCxList(
    // CFMX
    strings.FindByName("*CFMX_COMPAT*"),
    // RCX
-   strings.FindByShortNames(new List<string>(){"*RC2*", "*RC4*", "*RC5*", "*ARCFOUR*", "*Blowfish*"}, false));
+   strings.FindByShortNames(new string[]{"*RC2*", "*RC4*", "*RC5*", "*ARCFOUR*", "*Blowfish*"}, false));

-CxList weakTypesDes = All.NewCxList();
-weakTypesDes.Add(weakTypes, des);
+CxList weakTypesDes = All.NewCxList(weakTypes, des);

 // get all the weak types as unknown references
 CxList weakTypeRefs = unknownRefs.FindAllReferences(weakTypesDes.GetAssignee());

-CxList weakKeys = All.NewCxList();
-weakKeys.Add(weakTypes, weakTypeRefs, des);
+CxList weakKeys = All.NewCxList(weakTypes, weakTypeRefs, des);

 CxList keyPairGeneratorInitialize = methods.FindByMemberAccess("KeyPairGenerator.initialize");

@@ -40,7 +37,7 @@
    keyPairGeneratorInitialize.FindByParameters(weakKeys));

 //support MD5 MD2 MD4 SHA1
-CxList md5 = strings.FindByShortNames(new List<string> {
+CxList md5 = strings.FindByShortNames(new string[] {
        "\"MD5\"",
        "\"MD2\"",
        "\"SHA-1\"",
@@ -51,7 +48,7 @@

 //digesUtils
 CxList digestUtilElements = methods.FindByMemberAccess("DigestUtils.*");
-CxList digestElements = digestUtilElements.FindByShortNames(new List<string> {
+CxList digestElements = digestUtilElements.FindByShortNames(new string[] {
        "md5*",
        "md2*",
        "sha1*"});
@@ -59,7 +56,7 @@

 //HMAC
 CxList hmacs = methods.FindByMemberAccess("MAC.getInstance");
-CxList hmacAlgorithms = strings.FindByShortNames(new List<string> {
+CxList hmacAlgorithms = strings.FindByShortNames(new string[] {
        "\"HmacMD5\"",
        "\"HmacMD2\"",
        "\"HmacSHA-1\""});
```

Java / Java_Low_Visibility / Use_Of_Hardcoded_Password

Code changes

```diff
---
+++
@@ -3,16 +3,15 @@
 CxList psw = Find_All_Passwords();
 CxList stringLiterals = Find_Strings();
 CxList passwordString = Find_Password_Strings();
+CxList methods = Find_Methods();

-CxList passAndStrings = All.NewCxList();
-passAndStrings.Add(passwordString, psw);
+CxList passAndStrings = All.NewCxList(passwordString, psw);

 // Find password in an initialization operation
 CxList pswInLSide = psw.FindByAssignmentSide(CxList.AssignmentSide.Left);
 CxList pswInLSideDecl = pswInLSide.FindByType<Declarator>();

-CxList strLiterals = All.NewCxList();
-strLiterals.Add(stringLiterals);
+CxList strLiterals = All.NewCxList(stringLiterals);
 strLiterals -= emptyString;
 strLiterals -= nullsString;

@@ -40,7 +39,7 @@
 initializedPassword -= notHdPass;

 // Find password in an "equals" operation
-CxList eq = All.FindByMemberAccess("String.equals");
+CxList eq = methods.FindByMemberAccess("String.equals");
 CxList equalsPassword = strLiterals.GetByAncs(eq * psw.GetMembersOfTarget());

 eq *= strLiterals.GetMembersOfTarget();
@@ -55,7 +54,6 @@
 CxList assignPassword = pswInLSide.GetAncOfType<AssignExpr>();
 assignPassword = litInRSide.GetByAncs(assignPassword);

-CxList methods = Find_Methods();
 CxList connection = methods.FindByShortName("getConnection");
 CxList connetionParam2 = All.GetParameters(connection, 2);

@@ -78,8 +76,7 @@
 // Get second parameter
 CxList PasswordAuthenticationParam1 = All.GetParameters(passwordAuthentication, 1);

-CxList relevantParams = All.NewCxList();
-relevantParams.Add(KerberosKeyParam1, connetionParam2, PasswordAuthenticationParam1);
+CxList relevantParams = All.NewCxList(KerberosKeyParam1, connetionParam2, PasswordAuthenticationParam1);

 // Sanitize by binaries such as "+" and by concatenate - could be concatenated with a non hard-coded key,
 // which is OK
@@ -87,10 +84,8 @@
 bin = bin.FindByShortName("");
 CxList concat = All.FindByShortName("concatenate", false);

-CxList sanitize = All.NewCxList();
-
 CxList undefinedMethods = methods - methods.FindAllReferences(All.FindDefinition(methods));
-sanitize.Add(bin, concat, undefinedMethods);
+CxList sanitize = All.NewCxList(bin, concat, undefinedMethods);

 // Add the parameter itself, or whatever is influencing it
 CxList paramsAffectedByString = (relevantParams * strLiterals);
```

Java / Java_Low_Visibility / Use_Of_Hardcoded_Password_In_Config

Code changes

```
---
+++
@@ -22,10 +22,14 @@
 CxList smallPassword = longAssigner.Filter(p => p.ShortName.Length < 25);

 CxList strLiterals = Find_Strings();
-strLiterals -= strLiterals.FindByShortName("");
-strLiterals -= Find_Empty_Strings();
-strLiterals -= Find_Null_String_Name();
-strLiterals -= All.FindByNames(new string[]{"true", "false"});
+
+CxList toRemove = All.NewCxList(
+   strLiterals.FindByShortName(""),
+   Find_Empty_Strings(),
+   Find_Null_String_Name(),
+   All.FindByNames(new string[]{"true", "false"}));
+
+strLiterals -= toRemove;

 result = (smallPassword * strLiterals).GetAssignee();
 result.Add(androidPasswordsInXML);
```

Java / Java_Low_Visibility / Use_of_Hard_coded_Security_Constants

Code changes

```
---
+++
@@ -1,5 +1,5 @@
 // Find all vulnerable commands - currently only buffRead
-CxList buffRead = All.FindByMemberAccess("BufferedReader.read*");
+CxList buffRead = Find_Methods().FindByMemberAccess("BufferedReader.read*");
 // The second parameter is the vulnerable one
 CxList buffReadParam2 = All.GetParameters(buffRead, 2);
 // Al integeres
```

Java / Java__Low__Visibility / Use__of__RSA__Algorithm__without__OAEP

Code changes

```
---
+++
@@ -3,7 +3,7 @@
 // This query finds cipher RSA cryptographic
 // Algorithm without OAEP padding

-CxList cipherInstance = All.FindByMemberAccess("Cipher.getInstance");
+CxList cipherInstance = Find_Methods().FindByMemberAccess("Cipher.getInstance");

 CxList stringLiterals = Find_Strings();
 CxList cipherRSA = stringLiterals.FindByShortName("RSA*");
```

Java / Java__Low__Visibility / Using__Referer__Field__for__Authentication

Code changes

```
---
+++
@@ -1,6 +1,6 @@
 ///
 CxList Referer = All.FindByName("\"Referer\"");
-CxList header = All.FindByMemberAccess("request.getHeader");
+CxList header = Find_Methods().FindByMemberAccess("request.getHeader");
 header = header.DataInfluencedBy(Referer);

 CxList ifStmt = Find_Ifs();
```

Java / Java__Low__Visibility / UTF7__XSS

Code changes

```
---
+++
@@ -1,7 +1,7 @@
 CxList UTF7 = Find_Strings().FindByName("UTF-7");

 CxList response = All.FindByName("*Response.setCharacterEncoding", false);
-response.Add(All.FindByMemberAccess("HttpServletResponse.setCharacterEncoding"));
+response.Add(Find_Methods().FindByMemberAccess("HttpServletResponse.setCharacterEncoding"));

 UTF7 = response.DataInfluencedBy(UTF7);
```

Java / Java__Medium__Threat / CGI__Stored__XSS

Code changes

```
---
+++
@@ -17,8 +17,7 @@
```

```
    //it was added here and not inside the sanitizers because read is removing it again and there is no intention to change
    //general query
-   CxList dbRead = All.NewCxList();
-   dbRead.Add(db, read);
+   CxList dbRead = All.NewCxList(db, read);

    result = dbRead.InfluencingOnAndNotSanitized(outputs, sanitize);
    result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Medium_Threat / Cleartext_Submission_of_Sensitive_Information

Code changes

```
---
+++
@@ -1,11 +1,10 @@
 // Get all sensitive data.
-CxList personalInfo = All.NewCxList();
-personalInfo.Add(Find_Personal_Info(), Find_Password_Info());
+CxList personalInfo = All.NewCxList(Find_Personal_Info(), Find_Password_Info());

 // Remove strings, since they might contain: "Enter password".
 // A potential problem is that it might also contain: "password is...", but then it's hardcoded,
 // and not really sensitive information.
-List<string> integersToNotExclude = new List<string>{"*salary*", "*ccnlimit*"};
+string[] integersToNotExclude = new string[]{"*salary*", "*ccnlimit*"};
 CxList strings = Find_Strings();

 //A ResourceBundle can access its stored data by passing a key in the ResourceBundle.getString("key") method
@@ -17,7 +16,7 @@

 personalInfo -= strings;
 personalInfo -= Find_Integers() - personalInfo.FindByShortNames(integersToNotExclude, false);
-personalInfo -= personalInfo.FindByShortNames(new List<string> {"*regex*", "*pattern*"}, false);
+personalInfo -= personalInfo.FindByShortNames(new string[] {"*regex*", "*pattern*"}, false);

 // Remove declarators that are null or have an empty string assigned to it from personalInfo
 CxList nullOrEmpty = Find_Null_String_Name();
```

Java / Java_Medium_Threat / DoS_by_Sleep

Code changes

```
---
+++
@@ -9,14 +9,13 @@

 CxList tooltipDelay = Find_Jsp_Tags().GetMembersOfTarget().FindByMemberAccess("tooltipDelay.*");
 tooltipDelay = methods.GetParameters(tooltipDelay);
-CxList delay = All.NewCxList();
-delay.Add(sleep, tooltipDelay);
```

```
+CxList delay = All.NewCxList(sleep, tooltipDelay);


 CxList scheduleExecutor = methods.FindByMemberAccess("ScheduledExecutorService.schedule");
 CxList scheduleExecutorParams = All.GetParameters(scheduleExecutor, 1) - parameters;


 //Sleep sanitization
-CxList sleepMethods = delay.Clone();
+CxList sleepMethods = All.NewCxList(delay);
 CxList sleepParameters = All.GetParameters(sleepMethods) - parameters;
 sleepParameters.Add(scheduleExecutorParams);
```

Java / Java_Medium_Threat / Download_of_Code_Without_Integrity_Check

Code changes

```
---
+++
@@ -10,8 +10,7 @@
 CxList userInputs = Find_Interactive_Inputs();
 CxList objects = Find_Object_Create();

-CxList unkRefsAndStrings = All.NewCxList();
-unkRefsAndStrings.Add(strings, unkRefs, indexers, methods, binaryExpr);
+CxList unkRefsAndStrings = All.NewCxList(strings, unkRefs, indexers, methods, binaryExpr);

 CxList comparisonMethods = methods.FindByShortName("equals");
 comparisonMethods.Add(binaryExpr.FindByShortNames(new string[]{"!=","=="}));
@@ -19,12 +18,11 @@
 /**************
    SANITIZERS
 ***************/
-CxList sanitizers = All.NewCxList();
 CxList hashes = stringHashes.GetAncOfType<Declarator>();
 CxList hashesAssign = stringHashes.GetAncOfType<AssignExpr>();
 hashes.Add(hashesAssign.CxSelectDomProperty<AssignExpr>(x => x.Left));

-sanitizers.Add(hashes, weakHashes);
+CxList sanitizers = All.NewCxList(hashes, weakHashes);

 CxList sanitizersInfluencingComparison = comparisonMethods.InfluencedBy(sanitizers);


@@ -43,8 +41,7 @@
 CxList outputsThirdParam = methods.FindByMemberAccess("*Class.forName").FindByNumberOfParameters(3);

 // outputs
-CxList outputs = All.NewCxList();
-outputs.Add(outputsFirstParam, outputsThirdParam);
+CxList outputs = All.NewCxList(outputsFirstParam, outputsThirdParam);
```

```
 /*************
    INPUTS
```

Java / Java_Medium_Threat / Excessive_Data_Exposure

Code changes

```
---
+++
@@ -1,10 +1,9 @@
 //Get fields with sensitive information
-CxList sensitive = All.NewCxList();
-sensitive.Add(Find_Personal_Info(), Find_Password_Info());
+CxList sensitive = All.NewCxList(Find_Personal_Info(), Find_Password_Info());
 sensitive = sensitive.FindByType<FieldDecl>();

 //Remove some safe keywords
-List<string> safeKeywords = new List<string> {"*checkbox*", "*label*"};
+string[] safeKeywords = new string[] {"*checkbox*", "*label*"};
 sensitive -= sensitive.FindByShortNames(safeKeywords, false);

 //Get classes with sensitive fields
@@ -41,12 +40,15 @@
 CxList jacksonLib = cxXPath.FindXmlNodesByLocalNameAndValue("*pom.xml", 2, "groupId", "com.fasterxml.jackson.dataformat");
 if(jacksonLib.Count > 0)
 {
-   //@JsonIgnore
-   sensitive -= Find_Jackson_JsonIgnore_Sanitizer();
-   //@JsonIgnoreProperties
-   sensitive -= Find_Jackson_JsonIgnoreProperties_Sanitizer();
-   //@JsonFilter
-   sensitive -= Find_Jackson_JsonFilter_Sanitizer();
+   CxList toRemove = All.NewCxList(
+       //@JsonIgnore
+       Find_Jackson_JsonIgnore_Sanitizer(),
+       //@JsonIgnoreProperties
+       Find_Jackson_JsonIgnoreProperties_Sanitizer(),
+       //@JsonFilter
+       Find_Jackson_JsonFilter_Sanitizer());
+
+   sensitive -= toRemove;

    //@JsonIgnoreType
    outputStmts -= Find_Jackson_JsonIgnoreType_Sanitizer();
```

Java / Java_Medium_Threat / External_Control_of_Critical_State_Data

Code changes

```
---
+++
@@ -8,12 +8,12 @@
```

```
                           on the permission
 Since we currently don't have control influence, we will "simulate" it by checking "if" statement conditions.
 */
-
+CxList methods = Find_Methods();
 // General variables
-CxList getCookies = All.FindByMemberAccess("request.getCookies");
+CxList getCookies = methods.FindByMemberAccess("request.getCookies");
 getCookies.Add(Find_CookieValue_Annotation());
 CxList input = Find_Interactive_Inputs() - getCookies;
-CxList permissions = All.FindByMemberAccess("Permissions.add");
+CxList permissions = methods.FindByMemberAccess("Permissions.add");
 CxList conditions = Find_Conditions();
```

Java / Java__Medium__Threat / Frameable__Login__Page

Code changes

```
---
+++
@@ -12,8 +12,7 @@

 CxList methods = Find_Methods();
 CxList members = Find_MemberAccess();
-CxList methodsAndMembers = All.NewCxList();
-methodsAndMembers.Add(methods, members);
+CxList methodsAndMembers = All.NewCxList(methods, members);
 CxList unknownRefs = Find_UnknownReference();
 CxList strings = Find_Strings();

@@ -39,12 +38,12 @@
    CxList httpServletResponsesInWrapper = httpServletResponses.GetByAncs(wrapper);
    if(httpServletResponsesInWrapper.Count > 0){
        CxList responsesInWrapperMembers = unknownRefs.FindAllReferences(httpServletResponsesInWrapper).GetMembersOfTarget();
-       CxList addHeaderMethods = responsesInWrapperMembers.FindByShortNames(new List<string>{"setHeader", "addHeader"});
+       CxList addHeaderMethods = responsesInWrapperMembers.FindByShortNames(new string[]{"setHeader", "addHeader"});
        //If the header is not added, by default framing is allowed.
        if(addHeaderMethods.Count == 0){
            result.Add(wrapper);
        }else{
-           CxList xFrameOption = strings.GetParameters(addHeaderMethods, 1).FindByShortNames(new List<string>{"ALLOW-ALL", "ALLOWALL"});
+           CxList xFrameOption = strings.GetParameters(addHeaderMethods, 1).FindByShortNames(new string[]{"ALLOW-ALL", "ALLOWALL"});
            //If the X-FRAME-OPTIONS header is set to ALLOW ALL, framing is allowed.
            if(xFrameOption.Count > 0){
                result.Add(xFrameOption);
```

Java / Java__Medium__Threat / Hardcoded__password__in__Connection__String

Code changes

```diff
---
+++
@@ -8,35 +8,37 @@
 sanitizers.Add(Find_CollectionAccesses());
 
 CxList hardcodedStringSanitizers = methods.FindByMemberAccess("ResultSet.*");
-List<string> safeMethods = new List<string>{ "getNString", "getString" };
+string[] safeMethods = new string[]{ "getNString", "getString" };
 sanitizers.Add(hardcodedStringSanitizers.FindByShortNames(safeMethods));
 
 // Only first parameters of get/setProperty methods are safe
-CxList propertyMethods = All.FindByMemberAccess("Properties.*");
-List<string> safePropertyParams = new List<string>{ "getProperty", "setProperty" };
-sanitizers.Add(All.GetParameters(propertyMethods.FindByShortNames(safePropertyParams), 0));
+CxList propertyMethods = methods.FindByMemberAccesses("Properties", new string[]{ "getProperty", "setProperty"});
+sanitizers.Add(All.GetParameters(propertyMethods, 0));
 
 // Find creation of connections or connection strings, influenced by password
-CxList createExpressions = objects.Clone();
+CxList createExpressions = All.NewCxList(objects);
 CxList openConnection = createExpressions.FindByShortName("*Connection");
 result = openConnection.InfluencedByAndNotSanitized(psw, sanitizers);
 
-
 // Find password in relevant DB connection class initialization. There are three cases:
 // 1. DataSource.getConnection(String user, String password)
-CxList dataSourceConn = All.FindByMemberAccess("DataSource.getConnection");
+CxList dataSourceConn = methods.FindByMemberAccess("DataSource.getConnection");
 // 2. DriverManager.getConnection(String url, String user, String password)
-CxList driverManagerConn = All.FindByMemberAccess("DriverManager.getConnection");
+CxList driverManagerConn = methods.FindByMemberAccess("DriverManager.getConnection");
 // 3.1 DriverManagerDataSource.setPassword(String password)
-CxList dmdsSetPassword = All.FindByMemberAccess("DriverManagerDataSource.setPassword");
+CxList dmdsSetPassword = methods.FindByMemberAccess("DriverManagerDataSource.setPassword");
 // 3.2 new DriverManagerDataSource(String url, String username, String password)
 CxList driverManagerDataSource = createExpressions.FindByShortName("DriverManagerDataSource");
 // 4. new SimpleDriverDataSource(Driver instance, String url, String user, String password)
 CxList simpleDriverDataSource = createExpressions.FindByShortName("SimpleDriverDataSource");
 
 // Some connection initialization methods
-CxList pswInitMethods = All.NewCxList();
-pswInitMethods.Add(dataSourceConn, driverManagerConn, dmdsSetPassword, driverManagerDataSource, simpleDriverDataSource);
+CxList pswInitMethods = All.NewCxList(
+   dataSourceConn,
+   driverManagerConn,
+   dmdsSetPassword,
+   driverManagerDataSource,
+   simpleDriverDataSource);
```

```
 // Password parameters in connection initialization methods
 CxList pswParamInMethod = All.NewCxList();
@@ -60,9 +62,13 @@

 // More sanitizers
 // Sanitize params that are not passwords in DB connection methods
-CxList notPswInMethod = All.NewCxList();
-notPswInMethod.Add(dataSourceConn, driverManagerConn, dmdsSetPassword,
-   driverManagerDataSource, simpleDriverDataSource);
+CxList notPswInMethod = All.NewCxList(
+   dataSourceConn,
+   driverManagerConn,
+   dmdsSetPassword,
+   driverManagerDataSource,
+   simpleDriverDataSource);
+
 CxList notPswParamInMethod = All.GetParameters(notPswInMethod);
 notPswParamInMethod -= pswParamInMethod;
 sanitizers.Add(notPswParamInMethod,
```

Java / Java_Medium_Threat / HttpOnlyCookies

Code changes

```
---
+++
@@ -11,7 +11,7 @@
 CxList strings = Find_Strings();
 CxList setCookie = strings.GetParameters(setHeaders, 0).FindByShortName("Set-Cookie");
 setHeaders = setHeaders.FindByParameters(setCookie);
-CxList httponlyall = strings.FindByShortNames(new List<string>{"*HttpOnly*", "*httpOnly*"});
+CxList httponlyall = strings.FindByShortNames(new string[]{"*HttpOnly*", "*httpOnly*"});
 CxList secondParam = All.GetParameters(setHeaders, 1);
 CxList allUnderParam = All.GetByAncs(secondParam);
 CxList ur = allUnderParam * Find_UnknownReference();
```

Java / Java_Medium_Threat / JSF_Managed_Bean_PII_Leak

Code changes

```
---
+++
@@ -1,5 +1,4 @@
-CxList personalInfo = All.NewCxList();
-personalInfo.Add(Find_Personal_Info(), Find_Password_Info());
+CxList personalInfo = All.NewCxList(Find_Personal_Info(), Find_Password_Info());
 CxList classWithApplicationScoped = Find_CustomAttribute().FindByCustomAttribute("ApplicationScoped").GetFathers();
 CxList allClassMembers = All.FindAllMembers(classWithApplicationScoped);
 CxList memberInfluenced = allClassMembers.InfluencedBy(personalInfo).GetLastNodesInPath();
```

Java / Java_Medium_Threat / JWT_Lack_Of_Expiration_Time

Code changes

```
---
+++
@@ -1,7 +1,6 @@
 CxList methods = Find_Methods();
 CxList jwtsBuildRef = methods.FindByMemberAccess("Jwts.builder");
-CxList sanitizedMethod = All.FindByMemberAccess("JwtBuilder.*")
-   .FindByShortName("setExpiration").GetTargetOfMembers();
+CxList sanitizedMethod = methods.FindByMemberAccess("JwtBuilder.setExpiration").GetTargetOfMembers();
 sanitizedMethod.Add(methods.FindByShortName("setExpiration"));

 CxList lastNodeFlowJwts = methods.FindByShortName("compact");
```

Java / Java_Medium_Threat / JWT_Sensitive_Information_Exposure

Code changes

```
---
+++
@@ -1,4 +1,4 @@
-List < string > sinkNames = new List<string> { "setClaims", "claim", "addClaims", "setPayload" };
+string[] sinkNames = new string[] { "setClaims", "claim", "addClaims", "setPayload" };

 CxList methods = Find_Methods();
 CxList jwtsBuildRef = methods.FindByMemberAccess("Jwts.builder");
```

Java / Java_Medium_Threat / JWT_Use_Of_Hardcoded_Secret

Code changes

```
---
+++
@@ -1,4 +1,3 @@
-CxList strings = Find_Strings();
 CxList methods = Find_Methods();

 CxList jwtsRef = methods.FindByMemberAccess("Jwts.builder").GetTargetOfMembers();
@@ -9,7 +8,7 @@
 CxList signWithParam = All.GetParameters(signWithMet);
 CxList parameterInMethod = signWithParam.FindByType("Key");

-CxList flows = parameterInMethod.DataInfluencedBy(strings)
+CxList flows = parameterInMethod.DataInfluencedBy(Find_Strings())
    .ReduceFlow(CxList.ReduceFlowType.ReduceSmallFlow);

 flows.Add(signWithParam.FindByType<StringLiteral>());
@@ -19,7 +18,7 @@
 // Sanitize flows were JWT is not being signed through signWith method
 attributesValue -= attributesValue.SanitizeCxList(All.GetParameters(signWithMet, 1).Contained(attributesValue, CxList.GetStartEndNodesType.AllNodes));

-CxList sanitizedMethods = All.FindByMemberAccess("Cipher.getInstance");
```

```
+CxList sanitizedMethods = methods.FindByMemberAccess("Cipher.getInstance");
 attributesValue -= attributesValue.IntersectWithNodes(sanitizedMethods);

 flows.Add(attributesValue);
```

Java / Java_Medium_Threat / Privacy_Violation

Code changes

```
---
+++
@@ -2,15 +2,13 @@
 // which is streamed to an output.
 CxList strings = Find_Strings();
 CxList integerLiteral = Find_IntegerLiterals();
-CxList literals = All.NewCxList();
-literals.Add(strings, integerLiteral);
+CxList literals = All.NewCxList(strings, integerLiteral);
 CxList nullLiteral = Find_NullLiteral();
 CxList typeRefs = Find_TypeRef();

 // Find names that are suspected to be personal info, e.g. String PASSWORD, Integer SSN
 // Remove string literals, such as x = "password"
-CxList personal_info = All.NewCxList();
-personal_info.Add(Find_Personal_Info(), Find_Password_Info());
+CxList personal_info = All.NewCxList(Find_Personal_Info(), Find_Password_Info());
 personal_info -= strings;

 // 1) Exclude variables that are all uppercase
@@ -20,8 +18,7 @@
 // 2) Exclude constants that are assigned a literal
 CxList constants = personal_info * Find_Constants();
 CxList allConstRef = personal_info.FindAllReferences(constants);
-CxList allConstRefOrigin = All.NewCxList();
-allConstRefOrigin.Add(allConstRef);
+CxList allConstRefOrigin = All.NewCxList(allConstRef);

 // Find all assignments of string or integer literals
 CxList ConstAssignedL = literals.FindByFathers(allConstRef.FindByType<Declarator>());
@@ -51,15 +48,13 @@
 // Remove from personal_info all references that were removed above
 personal_info -= (allConstRefOrigin - allConstRef);

-CxList inputs = All.NewCxList();
-inputs.Add(
+CxList inputs = All.NewCxList(
    Find_DB_Out(),
    Find_Environment_Inputs(),
    Find_HTTP_Response_Read(),
    Find_Inputs(),
    Find_Local_Console_Inputs(),
-   Find_Portlets_Inputs()
-   );
+   Find_Portlets_Inputs());

 // We must find responses with sensitive types
 //Get classes with sensitive fields
@@ -71,12 +66,10 @@
 CxList sensitiveTypeUsage = Find_UnknownReference().FindAllReferences(sensitiveTypeDecls);
 CxList sensitiveResponse = sensitiveTypeUsage * Find_HTTP_Responses();

-CxList private_info = All.NewCxList();
-private_info.Add(
+CxList private_info = All.NewCxList(
    personal_info.DataInfluencedBy(inputs).GetLastNodesInPath(),
    personal_info * inputs,
-   sensitiveResponse
-   );
+   sensitiveResponse);

 personal_info = All.NewCxList(private_info);

@@ -95,13 +88,12 @@
 );

 // Define sanitize
-CxList sanitize = Find_DB();
-sanitize.Add(Find_Encrypt(), Find_UnitTest_Code(), Find_HashSanitize());
+CxList sanitize = All.NewCxList(Find_DB(), Find_Encrypt(), Find_UnitTest_Code(), Find_HashSanitize());

 // Add additional "integer" sanitizers
-sanitize.Add(All.FindByShortNames(new List<string> {"size", "length", "Index*", "indexOf"}, false),
+sanitize.Add(All.FindByShortNames(new string[] {"size", "length", "Index*", "indexOf"}, false),
    All.FindByName("*boolean.class.cast", StringComparison.OrdinalIgnoreCase),
-   All.FindByMemberAccess("Boolean.parse*"));
+   Find_Methods().FindByMemberAccess("Boolean.parse*"));

 // Split personal_info into variables and constants
 CxList variableRef = personal_info - allConstRef;
```

Java / Java_Medium_Threat / Process_Control

Code changes

```
---
+++
@@ -1,3 +1 @@
-CxList loadLibrary = Find_LoadLibrary();
-
-result = loadLibrary;
```

```
+result = Find_LoadLibrary();
```

Java / Java_Medium_Threat / ReDoS_In_Pattern

Code changes

```
---
+++
@@ -1,20 +1,20 @@
 CxList evilStrings = Find_Evil_Strings();
 CxList inputs = Find_Interactive_Inputs();
+CxList methods = Find_Methods();

 // Find all regex commands
-CxList regex = All.FindByMemberAccess("Pattern.compile");
+CxList regex = methods.FindByMemberAccess("Pattern.compile");

 // Find regex commands that are influenced by evil strings
 CxList activeEvilRegexes = evilStrings.DataInfluencingOn(regex);

 // Find all matches/splits of regexes
-CxList match = All.FindByMemberAccess("Matcher.matches");
+CxList match = methods.FindByMemberAccess("Matcher.matches");
 match = match.DataInfluencedBy(inputs);
-CxList split = All.FindByMemberAccess("Pattern.split");
+CxList split = methods.FindByMemberAccess("Pattern.split");
 split = split.DataInfluencedBy(inputs);

-CxList matchSplit = All.NewCxList();
-matchSplit.Add(match, split);
+CxList matchSplit = All.NewCxList(match, split);

 // Find relevant matches
 result = activeEvilRegexes.DataInfluencingOn(matchSplit);
```

Java / Java_Medium_Threat / Reliance_on_Cookies_without_Validation

Code changes

```
---
+++
@@ -12,7 +12,7 @@
 // General variables
 CxList getCookies = Find_GetCookies();

-CxList permissions = All.FindByMemberAccess("Permissions.add");
+CxList permissions = Find_Methods().FindByMemberAccess("Permissions.add");
 CxList conditions = Find_Conditions();
```

Java / Java_Medium_Threat / Same_Seed_in_PRNG

Code changes

```
---
+++
@@ -18,12 +18,10 @@
 CxList bin = Find_BinaryExpr();
 CxList numberAffecting = integers.InfluencingOnAndNotSanitized(setSeedParams, bin);

-CxList seedList = All.NewCxList();
-seedList.Add(numberSetSeed, pathsFromFinal, numberAffecting);
+CxList seedList = All.NewCxList(numberSetSeed, pathsFromFinal, numberAffecting);

 // We are only interested in random numbers which exert an influence on cryptographic and authentication methods
-CxList cryptoList = All.NewCxList();
-cryptoList.Add(Find_Encrypt(), Find_HashSanitize());
+CxList cryptoList = All.NewCxList(Find_Encrypt(), Find_HashSanitize());

 //We want as much information regarding the flow as possible
 result = seedList.InfluencingOn(cryptoList);
```

Java / Java_Medium_Threat / SSL_Verification_Bypass

Code changes

```
---
+++
@@ -47,14 +47,13 @@
    GetByAncs(hostVerifyList);

 // Newer version of apache library
-CxList hostVulnTypes = All.NewCxList();
-hostVulnTypes.Add(memberAccessList, unknRefList, paramList);
+CxList hostVulnTypes = All.NewCxList(memberAccessList, unknRefList, paramList);
 CxList hostAllList = hostVulnTypes.FindByShortNames(new string[]{"*AllowAllHostnameVerifier*","*NoopHostnameVerifier*"}, false).GetByAncs(methods);

 // No validation
 CxList newCerts = certList.FindByAssignmentSide(CxList.AssignmentSide.Left);

-List<string> validateMethodNames = new List<string> {
+string[] validateMethodNames = new string[] {
        "verify*",
        "checkValidity",
        "equals",
@@ -108,8 +107,7 @@
    .InfluencedBy(All.FindByType("KeyPair"))
    .GetLastNodesInPath();

-CxList selfSigned = All.NewCxList();
-selfSigned.Add(
+CxList selfSigned = All.NewCxList(
    selfSignedMthds.GetAssignee(),
    selfSignedMthds.GetFathers().GetAssignee(),
    keyPairInfluencedGoodCerts);
@@ -120,8 +118,7 @@

 //Find instances of TrustSelfSignedStrategy and TrustAllStrategy that are
 //passed as parameters to loadTrustMaterial
-CxList vulnParamTypes = All.NewCxList();
-vulnParamTypes.Add(objCreationList, unknRefList, memberAccessList);
+CxList vulnParamTypes = All.NewCxList(objCreationList, unknRefList, memberAccessList);
 CxList stratVulnType = vulnParamTypes.FindByTypes(new string[]{"TrustSelfSignedStrategy", "TrustAllStrategy"});

 result.Add(stratVulnType.GetAncOfType<MethodInvokeExpr>());
```

Java / Java_Medium_Threat / SSRF

Code changes

```
---
+++
@@ -5,8 +5,15 @@
 CxList argsInputs = All.GetParameters(mainDeclarations);
 inputs -= argsInputs;

+CxList sanitizers = Find_Remote_Requests_Sanitize();
+
 CxList requests = Find_Remote_Requests();
-CxList sanitizers = Find_Remote_Requests_Sanitize();
-
+
+// Exclude javax email methods from SSRF checks; they don't trigger network requests,
+// a prerequisite for SSRF vulnerabilities.
+string[] nonSSRFMethods = new string [] {"message.setText", "message.setSubject"};
+CxList nonSSRFMethodCalls = Find_Methods().FindByMemberAccesses(nonSSRFMethods);
+sanitizers.Add(nonSSRFMethodCalls);
+
 result = requests.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlowByPragma();
 result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Medium_Threat / Use_of_a_One_Way_Hash_without_a_Salt

Code changes

```
---
+++
@@ -4,7 +4,7 @@

 CxList sinks = Find_Digest_Commands();

-CxList possibleSalts = unkRefs.FindByShortNames(new List<string>{"*salt*", "*nonce*"});
+CxList possibleSalts = unkRefs.FindByShortNames(new string[]{"*salt*", "*nonce*"});
 CxList sanitizers = sinks.DataInfluencedBy(possibleSalts).GetLastNodesInPath();
 sanitizers.Add(Find_Password_Hash_Sanitize());
```

Java / Java_Spring / Spring_CSRF

Code changes

```
---
+++
@@ -1,8 +1,6 @@
-CxList methods = Find_Methods();
-
 //XML Files
 CxList csrfDisabled = cxXPath.FindXmlNodesByLocalName("*.xml", 2, "csrf", true, "disabled", "true", false, true);

-CxList disabled = methods.FindByMemberAccesses(new string[] {"csrf.disable", "CsrfConfigurer.disable"});
+CxList disabled = Find_Methods().FindByMemberAccesses(new string[] {"csrf.disable", "CsrfConfigurer.disable"});

 result.Add(csrfDisabled, disabled);
```

Java / Java_Spring / Spring_Missing_Content_Security_Policy

Code changes

```
---
+++
@@ -4,12 +4,11 @@
 if(springImports.Count > 0){

    CxList methods = Find_Methods();
-   CxList headers = All.NewCxList();

    //CSP inside meta
    CxList filesMetaCSP = cxXPath.FindXmlNodesByLocalName("*.xml", 2, "meta", true, "http-equiv",
        "Content-Security-Policy", false, true);
-   headers = filesMetaCSP.Clone();
+   CxList headers = All.NewCxList(filesMetaCSP);

    //CSP Tag
    CxList filesCSPTag = cxXPath.FindXmlNodesByLocalName("*.xml", 2, "content-security-policy", true);
```

Java / Java_Spring / Spring_Missing_XSS_Protection_Header

Code changes

```
---
+++
@@ -1,8 +1,6 @@
-CxList methods = Find_Methods();
-
 //XML Files
 CxList headersDisabledInXML = cxXPath.FindXmlNodesByLocalName("*.xml", 2, "xss-protection", true, "disabled", "true", false, true);

-CxList disabled = methods.FindByMemberAccess("xssProtection.disable");
```

```
+CxList disabled = Find_Methods().FindByMemberAccess("xssProtection.disable");


 result.Add(headersDisabledInXML, disabled, Find_Spring_DisabledDefaultHeaders());
```

Java / Java_Spring / Spring_Missing_X_Content_Type_Options

Code changes

```
---
+++
@@ -1,8 +1,6 @@
-CxList methods = Find_Methods();
-
 //XML Files
 CxList headersDisabledInXML = cxXPath.FindXmlNodesByLocalName("*.xml", 2, "content-type-options", true, "disabled", "true", false, true);

-CxList disabled = methods.FindByMemberAccess("contentTypeOptions.disable");
+CxList disabled = Find_Methods().FindByMemberAccess("contentTypeOptions.disable");

 result.Add(headersDisabledInXML, disabled, Find_Spring_DisabledDefaultHeaders());
```

Java / Java_Spring / Spring_Missing_X_Frame_Options

Code changes

```
---
+++
@@ -1,8 +1,6 @@
-CxList methods = Find_Methods();
-
 //XML Files
 CxList headersDisabledInXML = cxXPath.FindXmlNodesByLocalName("*.xml", 2, "frame-options", true, "disabled", "true", false, true);

-CxList disabled = methods.FindByMemberAccess("frameOptions.disable");
+CxList disabled = Find_Methods().FindByMemberAccess("frameOptions.disable");

 result.Add(headersDisabledInXML, disabled, Find_Spring_DisabledDefaultHeaders());
```

Java / Java_Spring / Spring_Use_of_Broken_or_Risky_Cryptographic_Primitive

Code changes

```
---
+++
@@ -1,7 +1,4 @@
-CxList methods = Find_Methods();
-CxList objectCreate = Find_Object_Create();
-
-List<string> deprecatedClassInSecure = new List<string>(){
+string[] deprecatedClassInSecure = new string[]{
        "StandardPasswordEncoder",
        "MessageDigestPasswordEncoder",
        "NoOpPasswordEncoder",
@@ -10,11 +7,11 @@
```

```
        "Md5PasswordEncoder",

        "PlaintextPasswordEncoder"};


-CxList deprecatedInstances = objectCreate.FindByShortNames(deprecatedClassInSecure);

+CxList deprecatedInstances = Find_Object_Create().FindByShortNames(deprecatedClassInSecure);


 string[] encryptorsClassInSecure = new string[]{

    "Encryptors.noOpText"};


-CxList encryptorsInstances = methods.FindByMemberAccesses(encryptorsClassInSecure);

+CxList encryptorsInstances = Find_Methods().FindByMemberAccesses(encryptorsClassInSecure);


 result.Add(deprecatedInstances, encryptorsInstances);
```

Java / Java_Spring / Spring_Use_Of_Hardcoded_Password

Code changes

```
---

+++

@@ -1,14 +1,10 @@

-CxList passwords = Find_All_Passwords();

-CxList customAttributes = Find_CustomAttribute();

-CxList methodDecls = Find_MethodDeclaration();

-CxList methods = Find_Methods();

-

 // Passwords in WebSecurityConfigurerAdapter.{configure,userDetailsService}

 CxList webSecurityCfgClasses = All.InheritsFrom("WebSecurityConfigurerAdapter");

-CxList webSecurityCfgMethods = methodDecls.FindByShortNames(new string[]{"configure", "userDetailsService"}).GetByAncs(webSecurityCfgClasses);

-CxList webSecurityCfgPasswords = methods.FindByShortName("password").GetByAncs(webSecurityCfgMethods);

+CxList webSecurityCfgMethods = Find_MethodDeclaration().FindByShortNames(new string[]{"configure", "userDetailsService"}).GetByAncs(webSecurityCfgClasses);

+CxList webSecurityCfgPasswords = Find_Methods().FindByShortName("password").GetByAncs(webSecurityCfgMethods);


 // Passwords in @Value

-CxList defaultValuePasswords = customAttributes.FindByShortName("Value").FindByFathers(passwords).GetFathers();

+CxList defaultValuePasswords = Find_CustomAttribute().FindByShortName("Value")

+   .FindByFathers(Find_All_Passwords()).GetFathers();


 result.Add(webSecurityCfgPasswords, defaultValuePasswords);
```

JavaScript / JavaScript_APISecurity / NodeJS_Express_WebApi_GetApiList

Code changes

```
---

+++

@@ -1,5 +1,11 @@

 CxList methodInvokes = Find_Methods();

 CxList unkRefs = Find_UnknownReference();

+CxList assignLeft = Find_Assign_Lefts();

+CxList arrayCreateExpr = Find_ArrayCreateExpr();

+
+CxList relevantUseParams = All.NewCxList();

+CxList relevantTypes = All.NewCxList();

+relevantTypes.Add(unkRefs, arrayCreateExpr, Find_String_Literal());


 string[] basicNodeMethods = new string[] {

    "get",

@@ -13,10 +19,42 @@

    "route"

    };


-CxList expressFramework = methodInvokes.FindByShortName("express");

-CxList allRefs = unkRefs.FindAllReferences(expressFramework.GetAssignee());

-CxList allRefsTargets = allRefs.GetMembersOfTarget();

-CxList endpoints = allRefsTargets.FindByShortNames(basicNodeMethods).FindByType<MethodInvokeExpr>();

-endpoints.Add(allRefsTargets.FindByShortName("use").FilterByDomProperty<MethodInvokeExpr>(x => x.Parameters.Count == 2));

+// Find: 'express' in var express = require("express");

+CxList express = Find_Require("express", 1);

+// Find: ' express()' in var app = express();

+CxList expressInit = methodInvokes.FindByShortName("express");

+// Find: 'router' in var router = express.Router();

+CxList createRouter = express.GetMembersOfTarget().FindByShortName("Router");

+// Find: 'router' in var router = require('express').Router();

+createRouter.Add(express.FindByShortName("Router"));

+CxList createRouterAssign = createRouter.FindByAssignmentSide(CxList.AssignmentSide.Right);

+CxList routers = (createRouter - createRouterAssign);

+routers.Add(assignLeft.FindByFathers(createRouterAssign.GetFathers()));

+// Find: 'app' in var app = express();

+CxList app = methodInvokes.FindAllReferences(expressInit).GetAssignee();

+// Find 'app' in app.get('/',func)

+CxList allRefs = unkRefs.FindAllReferences(app);

+// Cases when there isn't an 'app' reference, just exist one file with the following code:

+// const express = require('express');

+// const router = express.Router();

+// router.get(...);

+if(allRefs.Count == 0)

+{

+   allRefs.Add(unkRefs.FindAllReferences(routers));

+}

+// Find: 'get' in app.get('/',func)

+CxList targets = allRefs.GetMembersOfTarget();

+CxList filteredTargets = targets.FindByShortNames(basicNodeMethods);

+//Find: 'use' in app.use();

+CxList use = targets.FindByShortName("use");

+//The following choose the relevant use endpoints:

+//1 - Find: 'require' in app.use(require('./routes/exampleRoutes')); - unkRef

+//2 - Find: 'test' in app.use(test); where 'const test = require('./routes/exampleRoutes');' - ArrayCreateExpr

+CxList unkRefsUseParams = relevantTypes.GetParameters(use, 0).NotInfluencedBy(Find_LambdaExpr());

+filteredTargets.Add(use.FindByParameters(unkRefsUseParams));
```

```
+//Find the entry point files for routing, like 'index.js'
+CxList entryPointFlows = targets.DataInfluencedBy(routers);
+CxList entryPointExpressFlows = filteredTargets.DataInfluencedBy(express);
+entryPointFlows.Add(entryPointExpressFlows);

-result.Add(NodeJS_WebApi_Express(endpoints, basicNodeMethods));
+result.Add(NodeJS_WebApi_Express(routers, entryPointFlows));
```

Lua / Lua_High_Risk / Command_Injection

Code changes

```
---
+++
@@ -30,10 +30,7 @@
 CxList sanitizers = All.NewCxList(Find_Output_Sanitizers());
 //Remove encoders
 sanitizers -= Find_Encoders();
-
-//Regex Validation
-sanitizers.Add(Find_RegexValidation());
-
+
 //Allowlist sanitization
 sanitizers.Add(Find_AllowList_Sanitizers());
```

Lua / Lua_High_Risk / Reflected_XSS_All_Clients

Code changes

```
---
+++
@@ -1,25 +1,3 @@
-CxList methods = Find_Methods();
-
 CxList inputs = Find_Interactive_Inputs();

-CxList sanitizers = Find_Output_Sanitizers();
-
-CxList outputs = All.NewCxList();
-
-CxList confFile = Find_OpenResty_Conf_File();
-CxList unsafeContentTypeConfHeader = Find_Expressions().FindByShortName("default_type").GetAssigner().
-   FindByShortNames("text/html", "application/xml").GetByAncs(confFile);
-
-CxList confContentTypeAncBlock = unsafeContentTypeConfHeader.GetAncOfType<BlockStmt>();
-CxList echoMethods = methods.FindByShortNames("CxEcho", "CxEcho_Duplicate").GetByAncs(confContentTypeAncBlock);
-CxList ngxMethods = methods.FindByMemberAccesses("ngx", new string[]{"say", "print"});
-
-if(unsafeContentTypeConfHeader.Count > 0)
-   outputs.Add(echoMethods, ngxMethods.GetByAncs(confContentTypeAncBlock));
```

```
-
-//If the Content-Type header is NOT defined with "text/html" or "application/xml" then the output is sanitized.
-CxList unsafeHeaders = Find_Unsafe_ContentType_Headers();
-if(unsafeHeaders.Count > 0)
-   outputs.Add(ngxMethods);
-
-result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = Find_XSS(inputs);
```

Lua / Lua__High__Risk / Stored__Command__Injection

Code changes

```
---
+++
@@ -31,9 +31,6 @@
 //Remove encoders
 sanitizers -= Find_Encoders();

-//Regex Validation
-sanitizers.Add(Find_RegexValidation());
-
 //Allowlist sanitization
 sanitizers.Add(Find_AllowList_Sanitizers());
```

Lua / Lua__High__Risk / Stored__XSS

Code changes

```
---
+++
@@ -1,25 +1,3 @@
-CxList methods = Find_Methods();
-
 CxList inputs = All.NewCxList(Find_Stored_Inputs_DB(), Find_Stored_Inputs_Files(), Find_Stored_Inputs_Caches());

-CxList sanitizers = Find_Output_Sanitizers();
-
-CxList outputs = All.NewCxList();
-
-CxList confFile = Find_OpenResty_Conf_File();
-CxList unsafeContentTypeConfHeader = Find_Expressions().FindByShortName("default_type").GetAssigner().
-   FindByShortNames("text/html", "application/xml").GetByAncs(confFile);
-
-CxList confContentTypeAncBlock = unsafeContentTypeConfHeader.GetAncOfType<BlockStmt>();
-CxList echoMethods = methods.FindByShortNames("CxEcho", "CxEcho_Duplicate").GetByAncs(confContentTypeAncBlock);
-CxList ngxMethods = methods.FindByMemberAccesses("ngx", new string[]{"say", "print"});
-
-if(unsafeContentTypeConfHeader.Count > 0)
-   outputs.Add(echoMethods, ngxMethods.GetByAncs(confContentTypeAncBlock));
-
```

```
-//If the Content-Type header is NOT defined with "text/html" or "application/xml" then the output is sanitized.
-CxList unsafeHeaders = Find_Unsafe_ContentType_Headers();
-if(unsafeHeaders.Count > 0)
-   outputs.Add(ngxMethods);
-
-result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = Find_XSS(inputs);
```

Lua / Lua_Low_Visibility / Command_Argument_Injection

Code changes

```
---
+++
@@ -19,9 +19,6 @@
 //Remove encoders
 sanitizers -= Find_Encoders();

-//Regex Validation
-sanitizers.Add(Find_RegexValidation());
-
 //Allowlist sanitization
 sanitizers.Add(Find_AllowList_Sanitizers());
```

Lua / Lua_Low_Visibility / Improper_Exception_Handling

Code changes

```
---
+++
@@ -145,8 +145,7 @@
 CxList errReturn = filterRelevantIndexerRefs.GetAssignee();
 CxList errReturnUnkRefs = Find_UnknownReference().FindAllReferences(errReturn);
 CxList relevantErrReturn = All.NewCxList(errReturn - Find_Declarators().FindDefinition(errReturnUnkRefs), errReturnUnkRefs);
-
-CxList relevantSinks = sinks.InfluencingOn(errReturn);
+relevantErrReturn = relevantErrReturn.FindByShortNames("err", "_");

 CxList sanitizers = methods.FindByShortName("xpcall");
 sanitizers.Add(errReturnUnkRefs.FindByFathers(Find_Ifs()));
```

Lua / Lua_Low_Visibility / Information_Exposure_Through_Server_Log

Code changes

```
---
+++
@@ -2,13 +2,17 @@
 CxList inputs = Find_Sensitive_Information();
 inputs.Add(inputs.GetAncOfType<IndexerRef>());

-// TODO: Missing logging level openaResty sanitizer
 CxList sanitizers = Find_Output_Sanitizers() - Find_Encoders();
 sanitizers.Add(inputs.InfluencingOn(sanitizers));


-//return inputs;

+// Logging level

+CxList loggingMethods = Find_Methods_By_Logging_Level();

+

 CxList ngxLog = Find_Methods().FindByMemberAccess("ngx.log");

+ngxLog -= loggingMethods.FindByMemberAccess("ngx.log");

+

 CxList errLogRaw = Find_Import_Refs_By_Package_Name("ngx.errlog").GetMembersOfTarget().FindByShortName("raw_log");

+errLogRaw -= loggingMethods.FindByShortName("raw_log");


 CxList sinks = allParams.GetByAncs(ngxLog);

 sinks = sinks - allParams.GetParameters(ngxLog, 0);
```

Lua / Lua_Low_Visibility / Insufficient_Session_Expiration

Code changes

```
---

+++

@@ -3,7 +3,7 @@

 CxList integerLiterals = Find_IntegerLiterals();


 CxList restySessionImport = Find_Import_Refs_By_Package_Name("resty.session").GetMembersOfTarget()

-   .FindByShortNames("open", "new", "init");

+   .FindByShortNames("open", "new", "init", "start", "logout", "destroy");


 CxList inputs = integerLiterals.FindByShortName("0");
```

Lua / Lua_Low_Visibility / JWT_No_Expiration_Time_Validation

Code changes

```
---

+++

@@ -1,12 +1,14 @@

 CxList jwtRequire = Find_Import_Refs_By_Package_Name("resty.jwt");

 CxList trueAbsValue = Find_True_Abstract_Value();

 CxList methods = Find_Methods();

+CxList fieldDecls = Find_FieldDecls();


 CxList inputs = All.NewCxList(Find_Interactive_Inputs(), Find_Remote_Inputs());

 CxList sinks = jwtRequire.GetMembersOfTarget().FindByShortNames("load_jwt", "verify", "verify_jwt_obj");


 CxList requireExpClaimTrue = trueAbsValue.GetAncOfType<FieldDecl>().FindByShortName("require_exp_claim");

-CxList sanitizers = All.NewCxList(methods.FindByParameters(requireExpClaimTrue), Find_JWT_Builtin_Validators());

+CxList sanitizers = All.NewCxList(methods.FindByParameters(requireExpClaimTrue), Find_JWT_Builtin_Validators().

+   FindByParameters(fieldDecls.FindByShortName("exp")));

 //e.g. local claim_spec = { require_exp_claim = true }

 sanitizers.Add(sinks.InfluencedBy(requireExpClaimTrue.GetAncOfType<Declarator>().GetLastNodesInPath()));
```

Lua / Lua_Low_Visibility / JWT_No_NotBefore_Validation

Code changes

```
---
+++
@@ -2,12 +2,14 @@
 CxList trueAbsValue = Find_True_Abstract_Value();
 CxList methods = Find_Methods();
 CxList unkRefs = Find_UnknownReference();
+CxList fieldDecls = Find_FieldDecls();

 CxList inputs = All.NewCxList(Find_Interactive_Inputs(), Find_Remote_Inputs());
 CxList sinks = jwtRequire.GetMembersOfTarget().FindByShortNames("load_jwt", "verify", "verify_jwt_obj");

 CxList requireNbfclaimTrue = trueAbsValue.GetAncOfType<FieldDecl>().FindByShortName("require_nbf_claim");
-CxList sanitizers = All.NewCxList(methods.FindByParameters(requireNbfclaimTrue), Find_JWT_Builtin_Validators());
+CxList sanitizers = All.NewCxList(methods.FindByParameters(requireNbfclaimTrue), Find_JWT_Builtin_Validators().
+   FindByParameters(fieldDecls.FindByShortName("nbf")));
 //e.g. local claim_spec = { require_nbf_claim = true }
 sanitizers.Add(sinks.InfluencedBy(requireNbfclaimTrue.GetAncOfType<Declarator>().GetLastNodesInPath()));
 CxList sanitizersIdxRefAss = Find_IndexerRefs().FindAllReferences(sanitizers.GetAssignee()).GetAssignee();
```

Lua / Lua_Low_Visibility / Missing_Framing_Policy

Code changes

```
---
+++
@@ -1,14 +1,24 @@
-CxList strings = Find_Strings();
-CxList xframeHeader = Find_Set_Header("X-Frame-Options");
+CxList strings = Find_String_Literal();
+
+CxList validXFO = strings.FindByShortNames("SAMEORIGIN", "DENY");
+CxList validCSP = strings.FindByShortName("frame-ancestors*")
+   - strings.FindByRegex(@"frame-ancestors(\s+http://[\w-*]+\.\w+)+");
+CxList validXfoAndCsp = All.NewCxList(validCSP, validXFO);
+CxList vulnerableHeaderValues = strings - validXfoAndCsp;
+
+CxList xFrameHeader = Find_Set_Header("X-Frame-Options");
 CxList cspHeader = Find_Set_Header("Content-Security-Policy");
+CxList sinksLua = All.NewCxList(xFrameHeader.GetAncOfType<IndexerRef>(), cspHeader.GetAncOfType<IndexerRef>());
+CxList sinksConf = All.NewCxList(xFrameHeader, cspHeader).InfluencedBy(vulnerableHeaderValues).GetLastNodesInPath();
+CxList sinks = All.NewCxList(sinksLua, sinksConf);

-CxList allowFromOrigin = strings.FilterByDomProperty<StringLiteral>(_ => _.Text.Equals("ALLOW-FROM origin"));
-CxList frameAncestorsNone = strings.FilterByDomProperty<StringLiteral>(_ => _.Text.Equals("frame-ancestors 'none';"));
-CxList frameAncestorsNoneNs = frameAncestorsNone.GetAncOfType<NamespaceDecl>();
-allowFromOrigin -= allowFromOrigin.GetByAncs(frameAncestorsNoneNs);
```

```
+CxList xFrameOptions = validXFO.GetAssignee();

+CxList contentSecurityPolicy = validCSP.GetAssignee();


-CxList fromAncestorsWildCard = strings.

-   FilterByDomProperty<StringLiteral>(_ => _.Text.Equals("default-src 'self'; frame-ancestors *"));

+CxList cleanHeaders = All.NewCxList(xFrameOptions, contentSecurityPolicy);

+sinks -= cleanHeaders;

+CxList cleanHeadersNamespaces = contentSecurityPolicy.GetAncOfType<NamespaceDecl>();

+sinks -= sinks.GetByAncs(cleanHeadersNamespaces);

+sinks -= sinks.FindByType<MemberAccess>();


-result.Add(xframeHeader.InfluencedBy(allowFromOrigin),

-   cspHeader.InfluencedBy(fromAncestorsWildCard));

+result = sinks;
```

Lua / Lua_Low_Visibility / Null_Pointer_Dereference

Code changes

```
---

+++

@@ -3,6 +3,8 @@

 CxList unknRefs = Find_UnknownReference();

 CxList membAccess = Find_MemberAccesses();

 CxList methods = Find_Methods();

+CxList exprs = Find_Expressions();

+CxList indexers = Find_IndexerRefs();


 //Inputs

 CxList assignValues = Find_Declarators().FindDescendantsOfType<Declarator>(variableDecls).GetAssigner();

@@ -19,11 +21,20 @@

    .FilterByDomProperty<AssociativeArrayExpr>(_ => _.RegularEntries.Count == 0);

 inputs.Add(emptyArray);


+CxList methodDecl = Find_MethodDecls();

+CxList returns = Find_ReturnStmt().GetByAncs(methodDecl);

+returns = exprs.FindDescendantsOfType<Expression>(returns);

+returns -= returns.FindByTypes(typeof(TupleCreateExpr), typeof(TupleInitializer));

+CxList methodDeclInvokes = methods.FindAllReferences(methodDecl);

+CxList methodsAssignees = indexers.FindAllReferences(methodDeclInvokes.GetAssignee()).GetAssignee();

+inputs.Add(methodsAssignees - methodsAssignees.InfluencedBy(returns).GetLastNodesInPath());

+

 //Sanitizers

 CxList sanitizers = assignExprs - inputs.GetAncOfType<AssignExpr>();

+sanitizers -= methodDeclInvokes.GetAncOfType<AssignExpr>();

+

 CxList conditions = Find_Conditions();

 CxList binaryExprs = Find_BinaryExpr();

-CxList exprs = Find_Expressions();

 CxList nullValues = Find_NullLiteral();
 nullValues.Add(nullAssign);


@@ -67,9 +78,10 @@
 //Sinks
 CxList inputsArray = inputs.FindByType<AssociativeArrayExpr>();
 CxList otherInputs = inputs - inputsArray;
-otherInputs = otherInputs.GetAssignee();
+otherInputs.Add(otherInputs.GetAssignee());
+
+otherInputs.Add(methodsAssignees.InfluencedByAndNotSanitized(inputs, sanitizers).GetLastNodesInPath());
 otherInputs -= otherInputs.FindByTypes(new Type[] {typeof(IndexerRef), typeof(MemberAccess)});
-
 inputsArray = unknRefs.FindAllReferences(inputsArray.GetAssignee());
 CxList sinks = inputsArray.GetMembersOfTarget().GetMembersOfTarget();


@@ -79,10 +91,11 @@
 inputsRefs -= inputsRefs.InfluencedByAndNotSanitized(toRemove, otherInputs).GetLastNodesInPath();
 sinks.Add(inputsRefs.GetMembersOfTarget());

-CxList indxs = Find_IndexerRefs().FindAllReferences(otherInputs);
+CxList indxs = indexers.FindAllReferences(otherInputs);
 CxList indxsRefs = unknRefs.FindDescendantsOfType<UnknownReference>(indxs);


 indxs -= indxsRefs.InfluencedBy(inputs.FindByType<AssociativeArrayExpr>()).GetLastNodesInPath().GetAncOfType<IndexerRef>();
+indxs -= conditionsValues.GetByAncs(trueStmts);


 CxList length = membAccess.FindByShortName("length");
 sinks -= sinks.GetByAncs(length);
```

Lua / Lua_Low_Visibility / Password_In_Comment

Code changes

```
---
+++
@@ -5,11 +5,13 @@
    @"|salasana|schluessel|schluesselwort|senha|sifre|wachtwoord|wagwoord|watchword|zugangswort|PAROLACHIAVE|PAROLA CHIAVE" +
    @"|PAROLECHIAVI|PAROLE CHIAVI|paroladordine|verschluesselt|sisma)[^\s]*)";


-CxList singleLineComment = All.FindByRegexExt(@"(?<=--)(?!.*\[\[)(?:(?!--).)*" + passwords + @"(?=[^\n\w]*[:={[-]|[^\n\w]+[""']\w).*", "*.lua", true
-   , System.Text.RegularExpressions.RegexOptions.Multiline);
+CxList singleLineComment = All.FindByRegexExt(@"(?<=--|#)(?!.*\[\[)(?:(?!--|#).)*" + passwords
+   + @"(?=[^\n\w]*[:={[-]|[^\n\w]+[""']\w).*", new List<string>{"*.lua", "*.conf"}, true,
+   CxList.CxRegexOptions.None, RegexOptions.Multiline | RegexOptions.IgnoreCase);


-CxList multiLineComment = All.FindByRegexExt(@"(?<=--\[\[)(?:(?!\]\]).)*?" + passwords + @"(?=\s*[:={[-]|\s+[""']\w).*?(?=\]\])", "*.lua", true
-   , System.Text.RegularExpressions.RegexOptions.Singleline);
+CxList multiLineComment = All.FindByRegexExt(@"(?<=--\[\[)(?:(?!\]\]).)*?" + passwords
+   + @"(?=\s*[:={[-]|\s+[""']\w).*?(?=\]\])", "*.lua", true
+   , RegexOptions.Singleline | RegexOptions.IgnoreCase);
```

```
 CxList passwordsInComments = All.NewCxList(singleLineComment, multiLineComment);
```

Lua / Lua_Low_Visibility / Reliance_on_DNS_Lookups_in_a_Decision

Code changes

```
---
+++
@@ -29,10 +29,5 @@
 allRefLastNode.Add(refers.FindAllReferences(allRefLastNodeIndxRefAss));

 CxList secondSink = allRefLastNode.DataInfluencingOn(condValues).GetLastNodesInPath();
-
-CxList sanitizers = Find_Unarys();
-sanitizers.Add(allRefLastNode.GetMembersOfTarget().FindByShortName("errcode"));

-CxList secondFlow = inputs.InfluencingOnAndNotSanitized(secondSink, sanitizers);
-
-result.Add(firstSink,secondFlow);
+result = inputs.InfluencingOn(secondSink);
```

Lua / Lua_Low_Visibility / Stored_Command_Argument_Injection

Code changes

```
---
+++
@@ -19,9 +19,6 @@
 //Remove encoders
 sanitizers -= Find_Encoders();

-//Regex Validation
-sanitizers.Add(Find_RegexValidation());
-
 //Allowlist sanitization
 sanitizers.Add(Find_AllowList_Sanitizers());
```

Lua / Lua_Low_Visibility / Using_Referer_Field_for_Authentication

Code changes

```
---
+++
@@ -20,7 +20,8 @@
 CxList loops = conditions.GetAncOfType<IterationStmt>();
 CxList conditionalStatements = All.NewCxList(ifs, loops);

-CxList session = Find_Import_Refs_By_Package_Name("resty.session").GetMembersOfTarget().FindByShortName("new").GetAssignee();
+CxList session = Find_Import_Refs_By_Package_Name("resty.session").GetMembersOfTarget()
+   .FindByShortNames("new", "open", "start").GetAssignee();
 session = Find_UnknownReference().FindAllReferences(session);
 CxList save = session.GetMembersOfTarget().FindByShortName("save");
```

Lua / Lua_Medium_Threat / DoS_by_Sleep

Code changes

```
---
+++
@@ -1,21 +1,27 @@
 CxList methods = Find_Methods();
 CxList integers = Find_IntegerLiterals();
 CxList intsAndUnkRefs = All.NewCxList(integers, Find_UnknownReference());
-CxList conditions = Find_Conditions().FilterByDomProperty<BinaryExpr>(_ => _?.Operator == BinaryOperator.LessThan ||
-   _?.Operator == BinaryOperator.LessThanOrEqual);
+CxList conditions = Find_Conditions();
+CxList inputs = All.NewCxList(Find_Interactive_Inputs(), Find_Stored_Inputs_DB());

 CxList intAndRealAbsValue = All.NewCxList(integers, Find_RealLiterals());
 CxList intAbsValue = intsAndUnkRefs.FindByAbstractValues(intAndRealAbsValue);

 conditions = conditions.InfluencedBy(intAbsValue);

-CxList inputs = All.NewCxList(Find_Interactive_Inputs(), Find_Stored_Inputs_DB());
+CxList lessThan = conditions.FilterByDomProperty<BinaryExpr>(_ => _?.Operator ==
+   BinaryOperator.LessThan || _?.Operator == BinaryOperator.LessThanOrEqual);
+CxList greaterThan = conditions.FilterByDomProperty<BinaryExpr>(_ => _?.Operator ==
+   BinaryOperator.GreaterThan || _?.Operator == BinaryOperator.GreaterThanOrEqual);
+
+CxList inputsLeft = lessThan.CxSelectDomProperty<BinaryExpr>(_ => _.Left)
+   .InfluencedBy(inputs).GetFirstNodesInPath();
+CxList inputsRight = greaterThan.CxSelectDomProperty<BinaryExpr>(_ => _.Right)
+   .InfluencedBy(inputs).GetFirstNodesInPath();

 CxList outputs = methods.FindByMemberAccess("ngx.sleep");

-CxList sanitizers = conditions.InfluencedBy(inputs).GetFirstNodesInPath();
-// e.g. if (ngx.var.arg_delay < 10) then
-//       ngx.sleep(ngx.var.arg_delay)
+CxList sanitizers = All.NewCxList(inputsLeft, inputsRight);
 sanitizers.Add(outputs.FindByParameters(inputs.FindAllReferences(sanitizers)));

 result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers);
```

Lua / Lua_Medium_Threat / DoS_from_Evil_Regex

Code changes

```
---
+++
@@ -1,9 +1,11 @@
 CxList userInputs = Find_Interactive_Inputs();
 CxList inputs = All.NewCxList(userInputs, Find_Stored_Inputs_DB());
 CxList unkRefs = Find_UnknownReference();
-
+CxList evilStrings = Find_Evil_Strings();
+CxList evilVars = evilStrings.GetAssignee();
+CxList evilParameters = All.NewCxList(evilStrings, unkRefs.FindAllReferences(evilVars));
 CxList outputs = Find_MemberAccesses().FindByMemberAccess("ngx.re").GetMembersOfTarget().
-   FindByShortNames("match", "gmatch", "find", "sub", "gsub");
+   FindByShortNames("match", "gmatch", "find", "sub", "gsub").FindByParameters(evilParameters);

 //Hardcoded Evil Regex pattern can't be from user input
 CxList invalidOutputPattern = userInputs.InfluencingOn(unkRefs.GetParameters(outputs, 1)).GetLastNodesInPath();
```

Lua / Lua_Medium_Threat / DoS_from_RegEx_Injection

Code changes

```
---
+++
@@ -6,7 +6,11 @@

 CxList stringMethodsSinks = unkRefs.FindByShortName("string").GetMembersOfTarget().
    FindByShortNames("find", "match", "gmatch", "gsub");
-CxList outputs = All.NewCxList(unkRefs.GetParameters(openRestySinks, 1), unkRefs.GetParameters(stringMethodsSinks, 1));
+CxList stringVars = Find_Strings().GetAssignee();
+CxList varMethodsSinks = unkRefs.FindAllReferences(stringVars).GetMembersOfTarget().
+   FindByShortNames("find", "match", "gmatch", "gsub");
+CxList outputs = All.NewCxList(unkRefs.GetParameters(openRestySinks, 1), unkRefs.GetParameters(stringMethodsSinks, 1),
+   unkRefs.GetParameters(varMethodsSinks, 1));

 CxList sanitizers = Find_AllowList_Sanitizers();
 CxList confFile = Find_OpenResty_Conf_File();
```

Lua / Lua_Medium_Threat / JWT_Use_Of_Hardcoded_Secret

Code changes

```
---
+++
@@ -1,3 +1,5 @@
+CxList exprs = Find_Expressions();
+
 string[] jwtSinksMethods = new string[]{
    "verify",
    "verify_jwt_obj",
@@ -9,6 +11,8 @@

 CxList strings = Find_String_Literal();

-CxList sinks = All.GetParameters(methods, 0);
+CxList sinks = exprs.GetParameters(methods, 0);
```

```
-result.Add(strings.InfluencingOnAndNotSanitized(sinks, Find_Methods()), sinks*strings);
+CxList sanitizers = exprs.GetParameters(Find_Methods()) - sinks;
+
+result.Add(strings.InfluencingOnAndNotSanitized(sinks, sanitizers), sinks*strings);
```

Lua / Lua_Medium_Threat / Misconfigured_HSTS_Header

Code changes

```
---
+++
@@ -2,7 +2,7 @@

 // Sinks = ngx header/add_header titled configurations
 CxList sinks = Find_MemberAccesses().FindByMemberAccess("ngx.header").GetFathers().FindByType<IndexerRef>();
-CxList confAddHeader = Find_UnknownReference().FindByShortName("add_header").FindByFathers(Find_AssignExpr());
+CxList confAddHeader = Find_Declarators().FindByShortName("add_header");
 sinks.Add(confFileElements * confAddHeader);

 // Sanitizers = ngx header/add_header titled configurations with max-age >= 31536000 and includeSubDomains configurations
@@ -12,7 +12,7 @@
 Regex subDomainsRegex = new Regex(@"^.*(includeSubDomains).*");

 // StringLiterals refering to the pretended header configurations
-CxList hstsConfigs = Find_String_Literal().FindByFathers(sinks.GetFathers());
+CxList hstsConfigs = sinks.GetAssigner().FindByType<StringLiteral>();

 foreach(CxList config in hstsConfigs) {
    // Actual configuration Text
```

Lua / Lua_Medium_Threat / Open_Redirect

Code changes

```
---
+++
@@ -1,5 +1,6 @@
 CxList memberAccesses = Find_MemberAccesses();
 CxList methods = Find_Methods();
+CxList decls = Find_Declarators();

 //Inputs
 CxList inputs = Find_Interactive_Inputs();
@@ -25,12 +26,54 @@
 // FP reduction
 CxList sinks = Find_Open_Redirect_Status(possibleSinks, statusCodes);

-//ngx.exit(302) without any url sanitization is a result by itself, but redundant if associated with another sink
+//ngx.exit(302) without any url sanitization is a result by itself when inside a location with a custom header,
+//but redundant if associated with another sink
 CxList exit = methods.FindByMemberAccess("ngx.exit").FindByParameters(statusCodes);
 CxList nonSinkedExits = exit * sinks;
```

```
+
+CxList confFile = Find_OpenResty_Conf_File();
+CxList locations = Find_MethodDecls().FindByShortName("location_*") * confFile;
+CxList addHeader = decls.FindByShortNames(new string[] {"add_header",
+   "more_clear_headers",
+   "more_set_headers",
+   "more_set_input_headers",
+   "more_clear_input_headers"
+});
+
+foreach(CxList location in locations){
+   if (addHeader.GetByAncs(location).Count > 0) {
+       CxList luaFileConfigs = decls.FindByShortName("*_by_lua_file").GetByAncs(location);
+
+       if(luaFileConfigs.Count > 0) {
+           foreach(CxList file in luaFileConfigs) {
+               string fileNamespace = file.CxSelectElementValues<Declarator,string>(_ => _.InitExpression.Text)[0].Split(' ')[1];
+
+               CxList namespaceImport = Find_Import().FilterByDomProperty<Import>(_ => _.Namespace.Equals(fileNamespace));
+
+               string fileName = namespaceImport.CxSelectElementValues<Import,string>(_ => _.ImportedFilename)[0];
+               if(fileName.EndsWith(".lua")) {
+                   string[] filePathNodes = fileName.Split('.');
+                   string title = filePathNodes[filePathNodes.Count() - 2];
+                   string extension = filePathNodes[filePathNodes.Count() - 1];
+                   string baseFileName = title + "." + extension;
+                   string path = fileName.Substring(0, fileName.IndexOf(baseFileName));
+                   path = path.Replace(".", "\\");
+                   fileName = "*" + path + baseFileName;
+               } else {
+                   string pathedFileName = fileName.Replace(".", "\\");
+                   fileName = "*" + pathedFileName + ".lua";
+               }
+               CxList fileExits = nonSinkedExits.FindByFileName(fileName);
+
+               if(fileExits.Count > 0) {
+                   result.Add(file.ConcatenatePath(fileExits));
+               };
+           }
+       }
+   };
+}

 //Sanitization
 CxList sainitizedUrl = Find_Open_Redirect_Sanitizers(sinks);

-result.Add(nonSinkedExits);
 result.Add(sinks.InfluencedByAndNotSanitized(inputs, sainitizedUrl));
```

Lua / Lua_Medium_Threat / Privacy_Violation

Code changes

```
---
+++
@@ -1,21 +1,13 @@
-CxList strings = Find_Strings();
-CxList unknowns = Find_UnknownReference();
-CxList methods = Find_Methods();
-CxList stringParamTypes = All.NewCxList(strings, unknowns);
+// Inputs (Private Info excluding email)
+CxList inputs = All.NewCxList(Find_General_Personal_Info(), Find_Business_Info());

-CxList inputs = All.NewCxList(Find_General_Personal_Info(), Find_Business_Info());
-inputs = inputs - inputs.FindByType<StringLiteral>();
-
+// Sinks
 CxList sinks = Find_Interactive_Outputs();

-CxList sanitizers = Find_Output_Sanitizers();
+// Sanitizers
+// Casting, Hashing and Encryption
+CxList sanitizers = All.NewCxList(Find_Hashing(), Find_Encryption(), Find_Boolean_Casting(), Find_Masking());
+sanitizers.Add(inputs.DataInfluencedBy(sanitizers).GetLastNodesInPath(), Find_Excessive_Data_Objects());

-CxList replaceMethods = methods.FindByShortName("gsub");
-CxList replaceSecondParameter = stringParamTypes.GetParameters(replaceMethods, 1);
-replaceSecondParameter -= replaceSecondParameter.FindByType<Param>();
-
-Regex maskingCharacters = new Regex(@"^\**");
-CxList sanitizingReplacement = replaceSecondParameter.FilterByDomProperty<StringLiteral>(x => maskingCharacters.IsMatch(x.Text));
-sanitizers.Add(inputs.DataInfluencedBy(sanitizingReplacement).GetLastNodesInPath());
-
+// Result
 result = inputs.InfluencingOnAndNotSanitized(sinks, sanitizers);
```

PHP / PHP_High_Risk / Deserialization_of_Untrusted_Data

Code changes

```
---
+++
@@ -5,10 +5,8 @@
 //remove all the deserializers in coditioned blocks where the condition is influenced by hash
 CxList allCond = Find_Conditions();

-CxList comparison = Find_BinaryExpr().GetByAncs(allCond).FindByShortNames(new []{"==", "!="});
-
-CxList relevantComparison = comparison.InfluencedBy(sanitizers);
-relevantComparison = relevantComparison.GetLastNodesInPath();
+CxList relevantComparison = Find_BinaryExpr().GetByAncs(allCond).FindByShortNames(new []{"==", "!="});
```

```
+relevantComparison = sanitizers.GetByAncs(relevantComparison);

 CxList relevantIfs = relevantComparison.GetAncOfType<IfStmt>();
 relevantIfs.Add(relevantComparison.GetAncOfType<IterationStmt>());
```

PHP / PHP_High_Risk / Stored_XPath_Injection

Code changes

```
---
+++
@@ -1,13 +1,12 @@
-CxList inputs = All.NewCxList();
-inputs.Add(Find_DB_Out(), Find_Read());
+CxList inputs = All.NewCxList(Find_DB_Out(), Find_Read());

 // Find_DB_Out and Find_Read might return unknown references on binds
 // For those, there is no lazy flow, so we need to perform heuristics:
 // 1. Add all references
 inputs.Add(Find_UnknownReference().FindAllReferences(inputs));
 // 2. Remove any that is sanitized
+// a better solution would be using flow, but that would take too much time
 CxList sanitizers = Find_XPath_Sanitize();
-CxList sanitized = inputs.InfluencedBy(sanitizers);
-inputs -= inputs.FindAllReferences(sanitized);
+inputs -= inputs.FindAllReferences(inputs.GetByAncs(sanitizers));

 result = Find_XPath_Injection(inputs);
```

Python / Python_Medium_Threat / Object_Access_Violation

Code changes

```
---
+++
@@ -4,14 +4,20 @@
 CxList inputs = Find_Inputs();
 CxList sanitize = Find_Sanitize();
 sanitize.Add(Find_Base64_Encode());
+
+//Remove RequestFactory inputs
+CxList requestFactory = Find_Methods_By_Import("django.test*", new string[]{"RequestFactory", "AsyncRequestFactory"});
+CxList requestFacInputs = inputs.GetTargetOfMembers().InfluencedBy(requestFactory).GetLastNodesInPath().GetMembersOfTarget();
+requestFacInputs.Add(requestFacInputs.GetAncOfType<IndexerRef>());
+inputs -= requestFacInputs;
+
 //The following are the attr methods that generate this vulnerability
 List<string> attrMethodNames = new List<string> {"getattr", "setattr", "hasattr", "delattr"};
 CxList attrMethods = methods.FindByShortNames(attrMethodNames);

-//If the second argument of hasattr is a string literal, is not vulnerable
-CxList hasattrMethods = attrMethods.FindByShortName("hasattr");
```

```
-CxList hasattrMethodsSndParam = stringLiterals.GetParameters(hasattrMethods, 1);

-CxList sanitizedMethods = hasattrMethods.FindByParameters(hasattrMethodsSndParam);

-attrMethods -= hasattrMethods;

+//If the second argument of the attr methods is a string literal, is not vulnerable

+CxList attrMethodsSndParam = stringLiterals.GetParameters(attrMethods, 1);

+CxList sanitizedMethods = attrMethods.FindByParameters(attrMethodsSndParam);

+attrMethods -= sanitizedMethods;


 result = attrMethods.InfluencedByAndNotSanitized(inputs, sanitize);
```

Scala / Scala__Medium__Threat / Use__of__a__One__Way__Hash__without__a__Salt

Code changes

```
---

+++

@@ -1,11 +1,19 @@

+CxList inputs = Find_Passwords();

+

+CxList outputs = Find_Digest_Commands();

+CxList updateCommands = All.FindByMemberAccess("MessageDigest.update");

+

 CxList unkRefs = Find_UnknownReference();


-CxList sources = Find_Passwords();

+CxList possibleSalts = unkRefs.FindByShortNames(new string[] {"*salt*", "*nonce*"});

+CxList updateWithSalt = updateCommands.DataInfluencedBy(possibleSalts).GetLastNodesInPath().GetTargetOfMembers();


-CxList sinks = Find_Digest_Commands();

+CxList sanitizedMsgDigestRefs = unkRefs.FindAllReferences(updateWithSalt);

+outputs -= outputs.DataInfluencedBy(possibleSalts).GetLastNodesInPath();


-CxList possibleSalts = unkRefs.FindByShortNames(new List<string>{"*salt*", "*nonce*"});

-CxList sanitizers = sinks.DataInfluencedBy(possibleSalts).GetLastNodesInPath();

-sanitizers.Add(Find_Password_Hash_Sanitize());

+CxList sanitizers = All.NewCxList();

+sanitizers.Add(

+   sanitizedMsgDigestRefs.GetMembersOfTarget(),

+   Find_Password_Hash_Sanitize());


-result = sinks.InfluencedByAndNotSanitized(sources, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

+result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Scala / Scala__Medium__Threat / Use__of__a__One__Way__Hash__with__a__Predictable__Salt

Code changes

```
---

+++

@@ -1,22 +1,42 @@

 CxList methods = Find_Methods();

 CxList unkRefs = Find_UnknownReference();
```

```
+CxList parameters = Find_Param();


 CxList sources = Find_Passwords();

+CxList updateSalts = methods.FindByShortName("update");


 CxList genSecretMethods = methods.FindByMemberAccess("SecretKeyFactory.generateSecret");

-CxList sinks = Find_Digest_Commands();

-sinks.Add(genSecretMethods);

+CxList digestMethods = Find_Digest_Commands();


+//Get only sinks influenced by random or hardcoded salts

 string[] randomMethods = new string[] {"Random.next*", "RandomUtils.next*"};

-CxList predRandom = All.GetParameters(methods.FindByMemberAccesses(randomMethods));

+CxList predRandom = parameters.GetParameters(methods.FindByMemberAccesses(randomMethods));

 predRandom.Add(unkRefs.FindByShortNames(new List<string>{"*salt*", "*nonce*"}));

+

+//1. salt used directly in SecretKeyFactory.generateSecret

+CxList sinks = genSecretMethods.InfluencedBy(predRandom).GetLastNodesInPath();

+

+//2. md.update(salt); md.digest();

+predRandom = predRandom.FindByFathers(parameters);

+CxList updateRandomSalts = updateSalts.InfluencedBy(predRandom).GetLastNodesInPath();

+CxList predRandomTargets = updateRandomSalts.GetTargetOfMembers();

+predRandomTargets = unkRefs.FindAllReferences(predRandomTargets).GetMembersOfTarget();

+sinks.Add(digestMethods * predRandomTargets);


 CxList safeRandom = methods.FindByMemberAccesses(new string[] {"SecureRandom.next*", "BytesKeyGenerator.generateKey"});

 CxList safeAsRandom = Find_ObjectCreations().FindByType("SecureRandom").GetAssignee().FindByType("Random");

 safeRandom.Add(unkRefs.FindAllReferences(safeAsRandom));

-sinks = sinks.DataInfluencedBy(predRandom).GetLastNodesInPath();

-sinks -= sinks.DataInfluencedBy(All.GetParameters(safeRandom));

+

+//salt variables generated by SecureRandom "influencing" MessageDigest objects

+CxList safeSaltParams = unkRefs.GetParameters(safeRandom);

+CxList safeSaltRefs = unkRefs.FindAllReferences(safeSaltParams);

+CxList updateSafeRandomSalts = updateSalts.InfluencedBy(safeSaltRefs).GetLastNodesInPath();

+CxList predSafeRandomTargets = updateSafeRandomSalts.GetTargetOfMembers();

+CxList safeSaltTargets = unkRefs.FindAllReferences(predSafeRandomTargets).GetMembersOfTarget();

+sinks -= sinks * safeSaltTargets;

+sinks -= sinks.InfluencedBy(safeSaltRefs).GetLastNodesInPath();


 CxList sanitizers = Find_Password_Hash_Sanitize();


-result = sinks.InfluencedByAndNotSanitized(sources, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

+result = sinks.InfluencedByAndNotSanitized(sources, sanitizers)

+.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Dart / Dart_Mobile_Medium_Threat / Relative_Path_Traversal

Code changes

```
---
+++
@@ -6,24 +6,26 @@
 // Inputs
 CxList inputs = Find_UI_Inputs();

+// Concats
+CxList concats = Find_BinaryExpr().GetByBinaryOperator(BinaryOperator.Add);
+
 // Sinks
-CxList sinks = Find_ObjectCreations().FindByShortNames("File", "Directory");
-sinks.Add(methods.FindByMemberAccesses(new string[]{"Uri.file", "Uri.directory"}));
+CxList allSinks = Find_Path_Traversal_Sinks();
+CxList sinks = allSinks.InfluencedBy(concats);
+
 // Separating sinks with concatenated strings in first argument because there's additional sanitizers for this case
 CxList sinksWithConcatenatedStrings = All.NewCxList();
 sinksWithConcatenatedStrings.Add(sinks.Where(sink => Find_BinaryExpr().GetParameters(sink, 0).Count != 0));
 sinks -= sinksWithConcatenatedStrings;

 // Sanitizers
-CxList commonSanitizers = methods.FindByShortNames("isAbsolute", "isRelative", "isWithin");
+CxList commonSanitizers = Find_Path_Traversal_Sanitizers();
 CxList concatenatedStringsSanitizers = methods.FindByShortNames("basename", "basenameWithoutExtension");
-commonSanitizers.Add(Find_WhiteListSanitizers());

 // Results
 CxList sanitizedInputs = inputs.InfluencingOn(commonSanitizers).GetTargetOfMembers();
 CxList sanitizedTargets = unknowns.FindAllReferences(sanitizedInputs);
 CxList allSanitizedInputs = inputs.GetMembersWithTargets(sanitizedTargets);
 CxList unsanitizedInputs = inputs - allSanitizedInputs;
-
 result = sinksWithConcatenatedStrings.InfluencedByAndNotSanitized(unsanitizedInputs, concatenatedStringsSanitizers);
 result.Add(sinks.InfluencedBy(unsanitizedInputs));
```

Lua / Lua_Low_Visibility / Privacy_Violation_in_JWT

Code changes

```
---
+++
@@ -1,9 +1,10 @@
-CxList inputs = Find_Sensitive_Information();
+//Private Info (excluding hardcoded values and email)
+CxList inputs = All.NewCxList(Find_General_Personal_Info(), Find_Business_Info());

-CxList jwtRequire = Find_Import_Refs_By_Package_Name("resty.jwt");
-CxList jwtCreate = jwtRequire.GetMembersOfTarget().FindByShortName("sign");
+CxList jwtSign = Find_Import_Refs_By_Package_Name("resty.jwt").GetMembersOfTarget().FindByShortName("sign");
+CxList sinks = Find_Expressions().GetParameters(jwtSign, 1);
```

```
-CxList encryption = Find_Encryption();

-CxList sanitizers = inputs.DataInfluencedBy(encryption).GetLastNodesInPath();

+CxList sanitizers = All.NewCxList(Find_Hashing(), Find_Encryption(), Find_Boolean_Casting(), Find_Masking());

+sanitizers.Add(inputs.DataInfluencedBy(sanitizers).GetLastNodesInPath(), Find_Excessive_Data_Objects());


-result = jwtCreate.InfluencedByAndNotSanitized(inputs, sanitizers);

+result = inputs.InfluencingOnAndNotSanitized(sinks, sanitizers);
```

Python / Python_Medium_Threat / ReDoS_Injection

Code changes