Version 9.6.6 - Queries Release Notes

New Queries:

Language	Group	Name	CWE
Rust	Rust_Best_Coding_Practice	Input_Path_Not_Canonicalized	73
Rust	Rust_Critical	Arbitrary_File_Write	669
Rust	Rust_High_Risk	Dangerous_File_Inclusion	829
Rust	Rust_High_Risk	LDAP_Injection	90
Rust	Rust_High_Risk	Observable_Timing_Discrepancy	0
Rust	Rust_High_Risk	Sensitive_Information_Exposure_in_Cleartext_Channel	319
Rust	Rust_High_Risk	Stored_LDAP_Injection	90
Rust	Rust_High_Risk	Unsafe_Archive_Unpacking	0
Rust	Rust_Low_Visibility	Cookie_Overly_Broad_Path	539
Rust	Rust_Low_Visibility	Divide_By_Zero	369
Rust	Rust_Low_Visibility	Frameable_Login_Page	829
Rust	Rust_Low_Visibility	Improper_Error_Handling	703
Rust	Rust_Low_Visibility	Improper_Transaction_Handling	460
Rust	Rust_Low_Visibility	Insecure_Value_of_the_SameSite_Cookie_Attribute	1275
Rust	Rust_Low_Visibility	Integer_Overflow	190
Rust	Rust_Low_Visibility	Log_Forging	117
Rust	Rust_Low_Visibility	Misconfigured_HSTS	346
Rust	Rust_Low_Visibility	Misconfigured_X_Content_Type_Options	693
Rust	Rust_Low_Visibility	Missing_Content_Security_Policy	284
Rust	Rust_Low_Visibility	Missing_Framing_Policy	1021
Rust	Rust_Low_Visibility	Missing_HSTS	346
Rust	Rust_Low_Visibility	PCI_Data_Exposure_in_Error_Messages	200
Rust	Rust_Low_Visibility	PCI_Data_Exposure_in_Files	200
Rust	Rust_Low_Visibility	PCI_Data_Exposure_in_Logs	200
Rust	Rust_Low_Visibility	PCI_Data_Exposure_in_URL	200
Rust	Rust_Low_Visibility	Permissive_Content_Security_Policy	346
Rust	Rust_Low_Visibility	Privacy_Violation_in_Error_Messages	200
Rust	Rust_Low_Visibility	Privacy_Violation_in_URL	200
Rust	Rust_Low_Visibility	Trust_Boundary_Violation_in_Session_Variables	501
Rust	Rust_Low_Visibility	Type_Conversion_Error	195
Rust	Rust_Low_Visibility	Unchecked_Return_Value_to_NULL_Pointer_Dereference	476
Rust	Rust_Low_Visibility	Use_of_Deprecated_API	477
Rust	Rust_Low_Visibility	Use_of_Non_Cryptographic_Random	330
Rust	Rust_Low_Visibility	Use_of_Unsafe_Keyword	242
Rust	Rust_Medium_Threat	HttpOnly_Cookie_Flag_Not_Set	1004
Rust	Rust_Medium_Threat	Insufficiently_Secure_Password_Storage_Algorithm_Parameters	522
Rust	Rust_Medium_Threat	Length_Extension_Attack	326
Rust	Rust_Medium_Threat	Parameter_Tampering	472
Rust	Rust_Medium_Threat	PCI_Data_Exposure	200
Rust	Rust_Medium_Threat	Permission_Manipulation_in_S3	285
Rust	Rust_Medium_Threat	Reliance_on_Reverse_DNS_Lookups_in_a_Decision	350
Rust	Rust_Medium_Threat	Secret_Leak	200
Rust	Rust_Medium_Threat	Secret_Leak_in_Error_Messages	200
Rust	Rust_Medium_Threat	Secret_Leak_in_Files	200

Language	Group	Name	CWE
Rust	Rust_Medium_Threat	Secret_Leak_in_Logs	200
Rust	Rust_Medium_Threat	Secret_Leak_in_URL	200
Rust	Rust_Medium_Threat	Secure_Cookie_Flag_Not_Set	614
Rust	Rust_Medium_Threat	Uncontrolled_Memory_Allocation	789
Rust	Rust_Medium_Threat	Unsafe_Reflection	470
Rust	Rust_Medium_Threat	Use_of_Cryptographically_Weak_PRNG	338
Rust	Rust_Medium_Threat	Using_Referer_Field_for_Authentication	287

Changed Queries:

Language	Group	Name	CWE	Changed Fields
ASP	ASP_Best_Coding_Practice	Hardcoded_Connection_String	798	**Source** has changed
ASP	ASP_Best_Coding_Practice	PersistSecurityInfo_is_True	0	**Source** has changed
ASP	ASP_High_Risk	Code_Injection	94	**Source** has changed
ASP	ASP_High_Risk	Command_Injection	77	**Source** has changed
ASP	ASP_High_Risk	Connection_String_Injection	99	**Source** has changed
ASP	ASP_High_Risk	Dangerous_File_Upload	434	**Source** has changed
ASP	ASP_High_Risk	LDAP_Injection	90	**Source** has changed
ASP	ASP_High_Risk	Resource_Injection	99	**Source** has changed
ASP	ASP_High_Risk	XPath_Injection	643	**Source** has changed
ASP	ASP_Low_Visibility	Hardcoded_password_in_Connection_String	547	**Source** has changed
ASP	ASP_Low_Visibility	Impersonation_Issue	520	**Source** has changed
ASP	ASP_Low_Visibility	Information_Exposure_Through_an_Error_Message	209	**Source** has changed
ASP	ASP_Low_Visibility	Insufficiently_Protected_Credentials	522	**Source** has changed
ASP	ASP_Low_Visibility	Log_Forging	117	**Source** has changed
ASP	ASP_Low_Visibility	Open_Redirect	601	**Source** has changed
ASP	ASP_Low_Visibility	Script_Poisoning	20	**Source** has changed
ASP	ASP_Low_Visibility	Thread_Safety_Issue	567	**Source** has changed
ASP	ASP_Low_Visibility	Trust_Boundary_Violation_in_Session_Variables	501	**Source** has changed
ASP	ASP_Low_Visibility	URL_Canonicalization_Issue	647	**Source** has changed
ASP	ASP_Medium_Threat	CSRF	352	**Source** has changed
ASP	ASP_Medium_Threat	DoS_by_Sleep	834	**Source** has changed
ASP	ASP_Medium_Threat	Parameter_Tampering	472	**Source** has changed
ASP	ASP_Medium_Threat	Path_Traversal	22	**Source** has changed
ASP	ASP_Medium_Threat	SQL_Injection_Evasion_Attack	89	**Source** has changed
ASP	ASP_Medium_Threat	Untrusted_Activex	618	**Source** has changed
ASP	ASP_Medium_Threat	Use_of_Hard_coded_Cryptographic_Key	321	**Source** has changed
Apex	Apex_Force_com_Code_Quality	Bulkify_Apex_Methods_Using_Collections_In_Methods	0	**Source** has changed
Apex	Apex_Force_com_Serious_Security_Risk	Cookies_Scoping	0	**Source** has changed
Apex	Apex_Force_com_Serious_Security_Risk	CRUD_Delete	472	**Source** has changed
Apex	Apex_Force_com_Serious_Security_Risk	CSRF	352	**Source** has changed
Apex	Apex_Force_com_Serious_Security_Risk	Dereferenced_Field	0	**Source** has changed
Apex	Apex_Force_com_Serious_Security_Risk	FLS_Create	285	**Source** has changed
Apex	Apex_Force_com_Serious_Security_Risk	FLS_Create_Partial	285	**Source** has changed
Apex	Apex_Force_com_Serious_Security_Risk	FLS_Read	285	**Source** has changed
Apex	Apex_Force_com_Serious_Security_Risk	FLS_Update	285	**Source** has changed
Apex	Apex_Force_com_Serious_Security_Risk	FLS_Update_Partial	285	**Source** has changed

Language	Group	Name	CWE	Changed Fields
Apex	Apex_Force_com_Serious_Security_Risk	Frame_Spoofing	79	**Source** has changed
Apex	Apex_Force_com_Serious_Security_Risk	inputText_Ignoring_FLS	0	**Source** has changed
Apex	Apex_Force_com_Serious_Security_Risk	Insecure_Cookie	614	**Source** has changed
Apex	Apex_Force_com_Serious_Security_Risk	Insecure_Endpoint	319	**Source** has changed
Apex	Apex_Force_com_Serious_Security_Risk	URL_Redirection_Attack	601	**Source** has changed
Apex	Apex_ISV_Quality_Rules	SOQL_Dynamic_null_in_Where	0	**Source** has changed
Apex	Apex_ISV_Quality_Rules	SOQL_with_All_Fields_in_Loop	400	**Source** has changed
Apex	Apex_ISV_Quality_Rules	SOSL_With_Where_Clause	0	**Source** has changed
Apex	Apex_Low_Visibility	Potential_URL_Redirection_Attack	601	**Source** has changed
Apex	Apex_Low_Visibility	Use_of_Broken_or_Risky_Cryptographic_Algorithm	327	**Source** has changed
Apex	Apex_Low_Visibility	Verbose_Error_Reporting	209	**Source** has changed
CPP	CPP_Buffer_Overflow	Buffer_Improper_Index_Access	129	**Source** has changed
CPP	CPP_Buffer_Overflow	Format_String_Attack	134	**Source** has changed
CPP	CPP_Low_Visibility	NULL_Pointer_Dereference	476	**Source** has changed
CPP	CPP_Low_Visibility	Unchecked_Array_Index	129	**Source** has changed
CPP	CPP_Medium_Threat	Divide_By_Zero	369	**Source** has changed
CPP	CPP_Medium_Threat	Pointer_Subtraction_Determines_Size	469	**Source** has changed
CSharp	CSharp_Best_Coding_Practice	Hardcoded_Connection_String	798	**Source** has changed
CSharp	CSharp_Best_Coding_Practice	PersistSecurityInfo_is_True	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_Low_Visibility	Heap_Inspection	244	**Source** has changed
CSharp	CSharp_Low_Visibility	Impersonation_Issue	520	**Source** has changed
CSharp	CSharp_Low_Visibility	Information_Leak_Through_Persistent_Cookies	539	**Source** has changed
CSharp	CSharp_Low_Visibility	Leaving_Temporary_Files	376	**Source** has changed
CSharp	CSharp_Low_Visibility	Open_Redirect	601	**Source** has changed
CSharp	CSharp_Low_Visibility	Reliance_on_DNS_Lookups_in_a_Decision	350	**Source** has changed
CSharp	CSharp_Low_Visibility	Stored_Command_Argument_Injection	88	**Source** has changed
CSharp	CSharp_Low_Visibility	Thread_Safety_Issue	567	**Source** has changed
CSharp	CSharp_Low_Visibility	Trust_Boundary_Violation_in_Session_Variables	501	**Source** has changed
CSharp	CSharp_Low_Visibility	URL_Canonicalization_Issue	647	**Source** has changed
CSharp	CSharp_Low_Visibility	Use_of_RSA_Algorithm_without_OAEP	780	**Source** has changed
CSharp	CSharp_Medium_Threat	CGI_XSS	79	**Source** has changed
CSharp	CSharp_Medium_Threat	Data_Filter_Injection	943	**Source** has changed
CSharp	CSharp_Medium_Threat	DoS_by_Sleep	834	**Source** has changed
CSharp	CSharp_Medium_Threat	Hardcoded_password_in_Connection_String	547	**Source** has changed
CSharp	CSharp_Medium_Threat	Improper_Restriction_of_XXE_Ref	611	**Source** has changed
CSharp	CSharp_Medium_Threat	Insecure_Cookie	614	**Source** has changed
CSharp	CSharp_Medium_Threat	Missing_Column_Encryption	311	**Source** has changed
CSharp	CSharp_Medium_Threat	MVC_View_Injection	74	**Source** has changed
CSharp	CSharp_Medium_Threat	Privacy_Violation	359	**Source** has changed
CSharp	CSharp_Medium_Threat	ReDoS_In_Validation	400	**Source** has changed
CSharp	CSharp_Medium_Threat	Session_Fixation	384	**Source** has changed
CSharp	CSharp_Medium_Threat	SSL_Verification_Bypass	599	**Source** has changed
CSharp	CSharp_Medium_Threat	SSRF	74	**Source** has changed
CSharp	CSharp_Medium_Threat	Stored_Command_Injection	77	**Source** has changed
CSharp	CSharp_Medium_Threat	Use_of_Cryptographically_Weak_PRNG	338	**Source** has changed

Language	Group	Name	CWE	Changed Fields
CSharp	CSharp_Medium_Threat	Use_of_Hard_coded_Cryptographic_Key	321	**Source** has changed
CSharp	CSharp_WebConfig	TraceEnabled	749	**Source** has changed
CSharp	CSharp_Windows_Phone	Client_Side_Injection	89	**Source** has changed
CSharp	CSharp_Windows_Phone	Hard_Coded_Cryptography_Key	321	**Source** has changed
CSharp	CSharp_Windows_Phone	Insecure_Data_Storage	312	**Source** has changed
CSharp	CSharp_Windows_Phone	Poor_Authorization_and_Authentication	287	**Source** has changed
CSharp	CSharp_Windows_Phone	Side_Channel_Data_Leakage	200	**Source** has changed
Cobol	Cobol_High_Risk	Command_Injection	77	**Source** has changed
Cobol	Cobol_High_Risk	Module_Injection	610	**Source** has changed
Cobol	Cobol_High_Risk	Reflected_XSS_All_Clients	79	**Source** has changed
Cobol	Cobol_High_Risk	Resource_Injection	99	**Source** has changed
Cobol	Cobol_High_Risk	Sql_Injection	89	**Source** has changed
Cobol	Cobol_Medium_Threat	Path_Traversal	22	**Source** has changed
Dart	Dart_Mobile_High_Risk	Resource_Updated_By_URL_Data	939	**Source** has changed
Dart	Dart_Mobile_Low_Visibility	Missing_Certificate_Pinning	295	**Source** has changed
Dart	Dart_Mobile_Low_Visibility	Private_Storage_WebView_JavaScript_Injection	79	**Source** has changed
Dart	Dart_Mobile_Low_Visibility	Self_SQL_Injection	89	**Source** has changed
Dart	Dart_Mobile_Low_Visibility	Self_WebView_JavaScript_Injection	79	**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	Absolute_Path_Traversal	36	**Source** has changed
Dart	Dart_Mobile_Medium_Threat	Insecure_Asymmetric_Cryptographic_Algorithm_Parameters	326	**Source** has changed
Dart	Dart_Mobile_Medium_Threat	Insecure_WebSocket_Connection	319	**Source** has changed
Dart	Dart_Mobile_Medium_Threat	Poor_Authorization_and_Authentication	287	**Source** has changed
Dart	Dart_Mobile_Medium_Threat	Public_Storage_SQL_Injection	89	**Source** has changed
Dart	Dart_Mobile_Medium_Threat	Public_Storage_WebView_JavaScript_Injection	79	**Source** has changed
Dart	Dart_Mobile_Medium_Threat	Relative_Path_Traversal	23	**Source** has changed
Dart	Dart_Mobile_Medium_Threat	SQL_Injection_from_URL_Scheme_or_Intent	89	**Source** has changed
Dart	Dart_Mobile_Medium_Threat	Use_of_Hardcoded_Cryptographic_IV	326	**Source** has changed
Dart	Dart_Mobile_Medium_Threat	Use_of_Hardcoded_Salt	760	**Source** has changed
Dart	Dart_Mobile_Medium_Threat	WebView_JavaScript_Injection_from_URL_Scheme	79	**Source** has changed
Go	Go_AWS_Lambda	AWS_Credentials_Leak	200	**Source** has changed
Go	Go_AWS_Lambda	Hardcoded_AWS_Credentials	798	**Source** has changed
Go	Go_AWS_Lambda	Unrestricted_Write_S3	639	**Source** has changed
Go	Go_AWS_Lambda	Use_of_Hardcoded_Cryptographic_Key_On_Server	321	**Source** has changed
Go	Go_High_Risk	Connection_String_Injection	99	**Source** has changed
Go	Go_High_Risk	Unsafe_Reflection	470	**Source** has changed
Go	Go_Low_Visibility	Empty_Password_In_Connection_String	521	**Source** has changed
Go	Go_Low_Visibility	Open_Redirect	601	**Source** has changed
Go	Go_Medium_Threat	Divide_By_Zero	369	**Source** has changed
Go	Go_Medium_Threat	Email_Content_Forgery	116	**Source** has changed
Go	Go_Medium_Threat	Hardcoded_Password_in_Connection_String	547	**Source** has changed
Go	Go_Medium_Threat	Insecure_Value_of_the_SameSite_Cookie_Attribute_in_Code	1275	**Source** has changed
Java	Java_Android	Client_Side_Injection	89	**Source** has changed
Java	Java_Android	Implicit_Intent_With_Read_Write_Permissions	668	**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

Language	Group	Name	CWE	Changed Fields
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	Poor_Authorization_and_Authentication	287	**Source** has changed
Java	Java_Android	Use_Of_Implicit_Intent_For_Sensitive_Communication	927	**Source** has changed
Java	Java_AWS_Lambda	AWS_Credentials_Leak	200	**Source** has changed
Java	Java_AWS_Lambda	DynamoDB_NoSQL_Injection	74	**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	Unrestricted_Delete_S3	639	**Source** has changed
Java	Java_AWS_Lambda	Unrestricted_Write_S3	639	**Source** has changed
Java	Java_AWS_Lambda	User_Based_SDK_Configurations	15	**Source** has changed
Java	Java_AWS_Lambda	Use_of_Hardcoded_Cryptographic_Key_On_Server	321	**Source** has changed
Java	Java_Best_Coding_Practice	Incorrect_Conversion_between_Numeric_Types	681	**Source** has changed
Java	Java_Best_Coding_Practice	Reliance_On_Untrusted_Inputs_In_Security_Decision	807	**Source** has changed
Java	Java_GWT	GWT_DOM_XSS	79	**Source** has changed
Java	Java_GWT	GWT_Reflected_XSS	79	**Source** has changed
Java	Java_High_Risk	Expression_Language_Injection_EL	917	**Source** has changed
Java	Java_High_Risk	Reflected_XSS_All_Clients	79	**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_High_Risk	Unsafe_JNDI_Lookup	20	**Source** has changed
Java	Java_High_Risk	Unsafe_Reflection	470	**Source** has changed
Java	Java_Low_Visibility	Authorization_Bypass_Through_User_Controlled_SQL_PrimaryKey	566	**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	Empty_Password_In_Connection_String	521	**Source** has changed
Java	Java_Low_Visibility	Exposure_of_System_Data	497	**Source** has changed
Java	Java_Low_Visibility	Heap_Inspection	244	**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_Comments	615	**Source** has changed
Java	Java_Low_Visibility	Information_Leak_Through_Persistent_Cookies	539	**Source** has changed
Java	Java_Low_Visibility	Information_Leak_Through_Shell_Error_Message	535	**Source** has changed
Java	Java_Low_Visibility	Insufficient_Session_Expiration	613	**Source** has changed
Java	Java_Low_Visibility	Log_Forging	117	**Source** has changed
Java	Java_Low_Visibility	Open_Redirect	601	**Source** has changed
Java	Java_Low_Visibility	Portability_Flaw_Locale_Dependent_Comparison	474	**Source** has changed
Java	Java_Low_Visibility	Reliance_on_DNS_Lookups_in_a_Decision	350	**Source** has changed
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_Medium_Threat	CGI_Reflected_XSS_All_Clients	79	**Source** has changed
Java	Java_Medium_Threat	Dangerous_File_Inclusion	829	**Source** has changed

Language	Group	Name	CWE	Changed Fields
Java	Java_Medium_Threat	Direct_Use_of_Unsafe_JNI	111	**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	External_Control_of_Critical_State_Data	642	**Source** has changed
Java	Java_Medium_Threat	External_Control_of_System_or_Config_Setting	15	**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	Input_Path_Not_Canonicalized	73	**Source** has changed
Java	Java_Medium_Threat	JSF_CSRF	352	**Source** has changed
Java	Java_Medium_Threat	JWT_Lack_Of_Expiration_Time	613	**Source** has changed
Java	Java_Medium_Threat	JWT_Use_Of_Hardcoded_Secret	798	**Source** has changed
Java	Java_Medium_Threat	Plaintext_Storage_of_a_Password	256	**Source** has changed
Java	Java_Medium_Threat	Privacy_Violation	359	**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	SSL_Verification_Bypass	599	**Source** has changed
Java	Java_Medium_Threat	SSRF	918	**Source** has changed
Java	Java_Medium_Threat	Unvalidated_Forwards	819	**Source** has changed
Java	Java_Medium_Threat	Use_of_Cryptographically_Weak_PRNG	338	**Source** has changed
Java	Java_Medium_Threat	Use_of_Native_Language	695	**Source** has changed
Java	Java_Medium_Threat	XQuery_Injection	652	**Source** has changed
Java	Java_Spring	Spring_Comparison_Timing_Attack	208	**Source** has changed
Java	Java_Spring	Spring_ModelView_Injection	74	**Source** has changed
Java	Java_Spring	Spring_Overly_Permissive_Cross_Origin_Resource_Sharing_Policy	346	**Source** has changed
Java	Java_Stored	Stored_Open_Redirect	601	**Source** has changed
Java	Java_Stored	Stored_XPath_Injection	643	**Source** has changed
JavaScript	JavaScript_Angular	Angular_Client_DOM_XSS	79	**Source** has changed
JavaScript	JavaScript_Angular	Angular_Client_Stored_DOM_XSS	79	**Source** has changed
JavaScript	JavaScript_AWS_Lambda	DynamoDB_NoSQL_Injection	74	**Source** has changed
JavaScript	JavaScript_AWS_Lambda	Unrestricted_Read_S3	639	**Source** has changed
JavaScript	JavaScript_AWS_Lambda	Unrestricted_Write_S3	639	**Source** has changed
JavaScript	JavaScript_AWS_Lambda	User_Based_SDK_Configurations	15	**Source** has changed
JavaScript	JavaScript_Cordova	Cordova_Code_Injection	94	**Source** has changed
JavaScript	JavaScript_Cordova	Cordova_File_Disclosure	538	**Source** has changed
JavaScript	JavaScript_Cordova	Cordova_File_Manipulation	552	**Source** has changed
JavaScript	JavaScript_Cordova	Cordova_Open_Redirect	601	**Source** has changed
JavaScript	JavaScript_High_Risk	Client_DOM_Stored_XSS	79	**Source** has changed
JavaScript	JavaScript_High_Risk	Client_Dynamic_File_Inclusion	829	**Source** has changed
JavaScript	JavaScript_High_Risk	Client_Resource_Injection	99	**Source** has changed
JavaScript	JavaScript_High_Risk	Prototype_Pollution	1321	**Source** has changed
JavaScript	Javascript_Kony	Kony_Code_Injection	94	**Source** has changed
JavaScript	Javascript_Kony	Kony_Hardcoded_EncryptionKey	321	**Source** has changed
JavaScript	Javascript_Kony	Kony_Path_Injection	73	**Source** has changed
JavaScript	Javascript_Kony	Kony_Second_Order_SQL_Injection	89	**Source** has changed
JavaScript	Javascript_Kony	Kony_SQL_Injection	89	**Source** has changed
JavaScript	Javascript_Kony	Kony_Stored_Code_Injection	94	**Source** has changed
JavaScript	Javascript_Kony	Kony_URL_Injection	601	**Source** has changed

Language	Group	Name	CWE	Changed Fields
JavaScript	Javascript_Kony	Kony_Use_WeakEncryption	326	**Source** has changed
JavaScript	Javascript_Kony	Kony_Use_WeakHash	328	**Source** has changed
JavaScript	Javascript_Lightning	Lightning_DOM_XSS	79	**Source** has changed
JavaScript	Javascript_Lightning	Lightning_Stored_XSS	79	**Source** has changed
JavaScript	JavaScript_Low_Visibility	Client_Cookies_Inspection	315	**Source** has changed
JavaScript	JavaScript_Low_Visibility	Client_Empty_Password	259	**Source** has changed
JavaScript	JavaScript_Low_Visibility	Client_HTML5_Easy_To_Guess_Database_Name	330	**Source** has changed
JavaScript	JavaScript_Low_Visibility	Client_Remote_File_Inclusion	829	**Source** has changed
JavaScript	JavaScript_Low_Visibility	Client_Weak_Password_Authentication	798	**Source** has changed
JavaScript	JavaScript_Low_Visibility	Information_Exposure_Through_Query_Strings	522	**Source** has changed
JavaScript	JavaScript_Low_Visibility	Insufficiently_Protected_Credentials	522	**Source** has changed
JavaScript	JavaScript_Medium_Threat	Client_DOM_Cookie_Poisoning	472	**Source** has changed
JavaScript	JavaScript_Medium_Threat	Client_HTML5_Information_Exposure	200	**Source** has changed
JavaScript	JavaScript_Medium_Threat	Client_HTML5_Insecure_Storage	312	**Source** has changed
JavaScript	JavaScript_Medium_Threat	Client_Potential_Code_Injection	94	**Source** has changed
JavaScript	JavaScript_Medium_Threat	Client_Potential_XSS	79	**Source** has changed
JavaScript	JavaScript_Medium_Threat	Client_ReDoS_From_Regex_Injection	400	**Source** has changed
JavaScript	JavaScript_Medium_Threat	Client_ReDoS_In_Replace	400	**Source** has changed
JavaScript	JavaScript_Medium_Threat	Client_Untrusted_Activex	618	**Source** has changed
JavaScript	JavaScript_Medium_Threat	Client_XPATH_Injection	643	**Source** has changed
JavaScript	JavaScript_Medium_Threat	Frameable_Login_Page	829	**Source** has changed
JavaScript	JavaScript_Medium_Threat	Unchecked_Input_For_Loop_Condition	606	**Source** has changed
JavaScript	JavaScript_Medium_Threat	XML_External_Entities_XXE	611	**Source** has changed
JavaScript	JavaScript_ReactNative	Clipboard_Information_Leakage	200	**Source** has changed
JavaScript	JavaScript_ReactNative	Insufficient_Transport_Layer_Security	319	**Source** has changed
JavaScript	JavaScript_Server_Side_Vulnerabilities	Cookie_Poisoning	472	**Source** has changed
JavaScript	JavaScript_Server_Side_Vulnerabilities	CSRF	352	**Source** has changed
JavaScript	JavaScript_Server_Side_Vulnerabilities	Dynamic_File_Inclusion	829	**Source** has changed
JavaScript	JavaScript_Server_Side_Vulnerabilities	Information_Exposure_Through_an_Error_Message	209	**Source** has changed
JavaScript	JavaScript_Server_Side_Vulnerabilities	JWT_Lack_Of_Expiration_Time	613	**Source** has changed
JavaScript	JavaScript_Server_Side_Vulnerabilities	MongoDB_NoSQL_Injection	89	**Source** has changed
JavaScript	JavaScript_Server_Side_Vulnerabilities	Open_Redirect	601	**Source** has changed
JavaScript	JavaScript_Server_Side_Vulnerabilities	Parameter_Tampering	472	**Source** has changed
JavaScript	JavaScript_Server_Side_Vulnerabilities	Stored_Code_Injection	94	**Source** has changed
JavaScript	JavaScript_Server_Side_Vulnerabilities	Stored_XSS	79	**Source** has changed
JavaScript	JavaScript_Server_Side_Vulnerabilities	Unrestricted_File_Upload	434	**Source** has changed
JavaScript	JavaScript_Server_Side_Vulnerabilities	Unsafe_Object_Binding	915	**Source** has changed
JavaScript	JavaScript_Server_Side_Vulnerabilities	Use_of_Broken_or_Risky_Cryptographic_Algorithm	327	**Source** has changed
JavaScript	JavaScript_Vue	Vue_DOM_XSS	79	**Source** has changed
JavaScript	JavaScript_XS	XS_Response_Splitting	113	**Source** has changed
JavaScript	JavasScript_Visualforce_Remoting	VF_Remoting_Client_Potential_Code_Injection	94	**Source** has changed
Lua	Lua_Medium_Threat	DoS_by_Sleep	834	**Source** has changed
Objc	ObjectiveC_High_Risk	Information_Exposure_Through_Extension	200	**Source** has changed
Objc	ObjectiveC_High_Risk	Reflected_XSS_All_Clients	79	**Source** has changed
Objc	ObjectiveC_High_Risk	Second_Order_SQL_Injection	89	**Source** has changed
Objc	ObjectiveC_High_Risk	Stored_XSS	79	**Source** has changed
Objc	ObjectiveC_High_Risk	Unsafe_Reflection	470	**Source** has changed

Language	Group	Name	CWE	Changed Fields
Objc	ObjectiveC_Low_Visibility	Allowed_Backup	530	**Source** has changed
Objc	ObjectiveC_Low_Visibility	Information_Exposure_Through_an_Error_Message	209	**Source** has changed
Objc	ObjectiveC_Low_Visibility	Information_Leak_Through_Response_Caching	524	**Source** has changed
Objc	ObjectiveC_Low_Visibility	Log_Forging	117	**Source** has changed
Objc	ObjectiveC_Low_Visibility	Plain_Text_Transport_Layer	311	**Source** has changed
Objc	ObjectiveC_Low_Visibility	Poor_Authorization_and_Authentication	287	**Source** has changed
Objc	ObjectiveC_Low_Visibility	Use_of_Hardcoded_Cryptographic_Key	321	**Source** has changed
Objc	ObjectiveC_Medium_Threat	Cut_And_Paste_Leakage	200	**Source** has changed
Objc	ObjectiveC_Medium_Threat	Insufficient_Transport_Layer_Input	319	**Source** has changed
PHP	PHP_High_Risk	Reflected_XSS	79	**Source** has changed
PHP	PHP_High_Risk	Stored_XSS	79	**Source** has changed
PHP	PHP_Medium_Threat	Hashing_Length_Extension_Attack	310	**Source** has changed
PHP	PHP_Medium_Threat	Header_Injection	113	**Source** has changed
PHP	PHP_Medium_Threat	Missing_Encryption_of_Sensitive_Data	311	DescriptionId changed from 3042 to **2182**
PHP	PHP_Medium_Threat	Open_Redirect	601	**Source** has changed
PHP	PHP_Medium_Threat	Value_Shadowing	233	**Source** has changed
Python	Python_AWS_Lambda	AWS_Credentials_Leak	0	**Source** has changed
Python	Python_AWS_Lambda	DynamoDB_NoSQL_Injection	74	**Source** has changed
Python	Python_AWS_Lambda	Hardcoded_AWS_Credentials	798	**Source** has changed
Python	Python_AWS_Lambda	Use_of_Hardcoded_Cryptographic_Key_On_Server	321	**Source** has changed
Python	Python_High_Risk	Command_Injection	77	**Source** has changed
Python	Python_High_Risk	LDAP_Injection	90	**Source** has changed
Python	Python_High_Risk	OS_Access_Violation	77	**Source** has changed
Python	Python_High_Risk	Second_Order_SQL_Injection	89	**Source** has changed
Python	Python_High_Risk	Unsafe_Deserialization	502	**Source** has changed
Python	Python_Low_Visibility	Django_Missing_Function_Level_Authorization	862	DescriptionId changed from 2167 to **2607**
Python	Python_Low_Visibility	Log_Forging	117	**Source** has changed
Python	Python_Low_Visibility	Marshmallow_Dumping_Without_Validation	1173	**Source** has changed
Python	Python_Low_Visibility	Trust_Boundary_Violation_in_Session_Variables	501	**Source** has changed
Python	Python_Medium_Threat	Django_Missing_Object_Level_Authorization	862	DescriptionId changed from 2167 to **2606**
Python	Python_Medium_Threat	DoS_by_Sleep	834	**Source** has changed
Python	Python_Medium_Threat	Filtering_Sensitive_Logs	532	**Source** has changed
Python	Python_Medium_Threat	Header_Injection	113	**Source** has changed
Python	Python_Medium_Threat	Improper_Restriction_of_XXE_Ref	611	**Source** has changed
Python	Python_Medium_Threat	Object_Access_Violation	610	**Source** has changed
Python	Python_Medium_Threat	ReDoS_Injection	400	**Source** has changed
RPG	RPG_Low_Visibility	Use_Of_Hardcoded_Password	259	**Source** has changed
Rust	Rust_Low_Visibility	Privacy_Violation_in_Files	538	**Source** has changed
Rust	Rust_Low_Visibility	Privacy_Violation_in_Logs	200	**Source** has changed
Rust	Rust_Medium_Threat	Empty_Password_In_Connection_String	521	**Source** has changed
Rust	Rust_Medium_Threat	Encoding_Used_Instead_of_Encryption	311	**Source** has changed
Rust	Rust_Medium_Threat	Privacy_Violation	359	**Source** has changed
Rust	Rust_Medium_Threat	SSRF	918	**Source** has changed
VbNet	VbNet_Best_Coding_Practice	Dynamic_SQL_Queries	89	**Source** has changed
VbNet	VbNet_Best_Coding_Practice	Hardcoded_Connection_String	798	**Source** has changed
VbNet	VbNet_Best_Coding_Practice	PersistSecurityInfo_is_True	0	**Source** has changed
VbNet	VbNet_High_Risk	Code_Injection	94	**Source** has changed

Language	Group	Name	CWE	Changed Fields
VbNet	VbNet_High_Risk	Command_Injection	77	**Source** has changed
VbNet	VbNet_High_Risk	Connection_String_Injection	99	**Source** has changed
VbNet	VbNet_High_Risk	Dangerous_File_Upload	434	**Source** has changed
VbNet	VbNet_High_Risk	Reflected_XSS_All_Clients	79	**Source** has changed
VbNet	VbNet_High_Risk	Resource_Injection	99	**Source** has changed
VbNet	VbNet_High_Risk	Second_Order_SQL_Injection	89	**Source** has changed
VbNet	VbNet_High_Risk	Stored_XSS	79	**Source** has changed
VbNet	VbNet_High_Risk	XPath_Injection	643	**Source** has changed
VbNet	VbNet_Low_Visibility	Impersonation_Issue	520	**Source** has changed
VbNet	VbNet_Low_Visibility	Information_Exposure_Through_an_Error_Message	209	**Source** has changed
VbNet	VbNet_Low_Visibility	Information_Leak_Through_Persistent_Cookies	539	**Source** has changed
VbNet	VbNet_Low_Visibility	Insufficiently_Protected_Credentials	522	**Source** has changed
VbNet	VbNet_Low_Visibility	Leaving_Temporary_Files	376	**Source** has changed
VbNet	VbNet_Low_Visibility	Log_Forging	117	**Source** has changed
VbNet	VbNet_Low_Visibility	Open_Redirect	601	**Source** has changed
VbNet	VbNet_Low_Visibility	Session_Clearing_Problems	613	**Source** has changed
VbNet	VbNet_Low_Visibility	Stored_Code_Injection	94	**Source** has changed
VbNet	VbNet_Low_Visibility	Thread_Safety_Issue	567	**Source** has changed
VbNet	VbNet_Low_Visibility	Trust_Boundary_Violation_in_Session_Variables	501	**Source** has changed
VbNet	VbNet_Low_Visibility	URL_Canonicalization_Issue	647	**Source** has changed
VbNet	VbNet_Low_Visibility	Use_of_Broken_or_Risky_Cryptographic_Algorithm	327	**Source** has changed
VbNet	VbNet_Medium_Threat	Buffer_Overflow	120	**Source** has changed
VbNet	VbNet_Medium_Threat	CGI_XSS	79	**Source** has changed
VbNet	VbNet_Medium_Threat	CSRF	352	**Source** has changed
VbNet	VbNet_Medium_Threat	Data_Filter_Injection	200	**Source** has changed
VbNet	VbNet_Medium_Threat	DoS_by_Sleep	834	**Source** has changed
VbNet	VbNet_Medium_Threat	Hardcoded_password_in_Connection_String	547	**Source** has changed
VbNet	VbNet_Medium_Threat	SQL_Injection_Evasion_Attack	89	**Source** has changed
VbNet	VbNet_Medium_Threat	Stored_Command_Injection	77	**Source** has changed
VbNet	VbNet_Medium_Threat	Stored_LDAP_Injection	90	**Source** has changed
VbNet	VbNet_Medium_Threat	Stored_XPath_Injection	643	**Source** has changed
VbNet	VbNet_Medium_Threat	Use_of_Hard_coded_Cryptographic_Key	321	**Source** has changed

Changed Source:

Apex / Apex_Force_com_Code_Quality / Bulkify_Apex_Methods_Using_Collections_In_Methods

Code changes

```
---
+++
@@ -3,8 +3,7 @@


 // Find db parameters
 CxList db = Find_DB_Output();
-CxList bareDB = All.NewCxList();
-bareDB.Add(db);
+CxList bareDB = All.NewCxList(db);
```

```
 CxList meth = All.FindAllReferences(db.GetAncOfType<MethodDecl>());


@@ -27,7 +26,8 @@
 methodsParams -= methodsParams.FindByTypes(new string[] {"list", "set", "map"});
 methodsParams -= methodsParams.FilterByDomProperty<ParamDecl>(x => x.Type.ArrayRanks != null);


-result = methodsParams.FindAllReferences(updateParams).DataInfluencingOn(updateWhat);
+result = methodsParams.FindAllReferences(updateParams).DataInfluencingOn(updateWhat)
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);


 // Add cases where we have flow problem
 updateParams = apexObjects.GetMembersOfTarget().GetByAncs(apex.GetParameters(db));
```

Apex / Apex_Force_com_Serious_Security_Risk / Cookies_Scoping

Code changes

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


 CxList testCode = Find_Test_Code();


-result = put.InfluencedByAndNotSanitized(influencingInput, testCode);
+result = put.InfluencedByAndNotSanitized(influencingInput, testCode)
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);


 result -= testCode;
```

Apex / Apex_Force_com_Serious_Security_Risk / CRUD_Delete

Code changes

```
---
+++
@@ -1,4 +1,8 @@
 CxList strings = Find_Strings();
+CxList typeReferences = Find_TypeRef();
+CxList genericTypeRefs = Find_GenericTypeRefs();
+CxList decls = Find_Declarators();
+CxList targetTypes = All.NewCxList(Find_Expressions(), Find_UnknownReference(), typeReferences);
 CxList findIfNot = Find_If_Not_Stmt();
 CxList findIfYes = Find_If_Yes_Stmt();
 CxList apexFiles = Find_Apex_Core_Code();
@@ -22,7 +26,7 @@
     methods.FindByShortName("getDescribe", false).GetMembersOfTarget().FindByShortName("isDeletable", false));
 isDeletable.Add(isSomewhatDeletable);


-CxList returnValues = deleteMembers;
+CxList returnValues = All.NewCxList(deleteMembers);
 CxList isDeletableCond = All.NewCxList();
 isDeletableCond.Add(isDeletable,
```

```
    Find_General_Permissions());
@@ -31,7 +35,6 @@
 {
     // Methods that call isDeletable
     CxList isDeletable2 = isDeletable - isDeletable.FindByShortName("isDeletable", false);
-


     CxList ifStmt = isDeletable.GetAncOfType<IfStmt>();


@@ -59,7 +62,8 @@
                 ).ToArray();


             returnValues -= potentialRemove.FindByTypes(names);
-            returnValues -= returnValues.FindAllReferences(apexConditions);
+            returnValues -= returnValues.FindAllReferences(apexConditions);
+
             foreach (CxList p in potentialRemove)
             {
                 CxList parameters = strings.GetByAncs(apexFiles.GetParameters(p));
@@ -97,11 +101,11 @@
     CxList sanitizedRefs = All.NewCxList();

     foreach ( CxList deletableType in deletableTypes){
-        CxList refs = All.FindByType(deletableType);
+        CxList refs = targetTypes.FindByType(deletableType);
         // We should assume generic types of deletable type sanitized as well
         CxList typeRefs = refs.FindByType<TypeRef>();
         CxList genericTypes = typeRefs.GetFathers().FindByType<GenericTypeRef>();
-        refs.Add(All.FindByType(genericTypes));
+        refs.Add(targetTypes.FindByType(genericTypes));


         CxList ifFather = deletableType.GetAncOfType<IfStmt>() * findIfYes;
         sanitizedRefs.Add(refs.GetByAncs(ifFather));
@@ -134,13 +138,14 @@
     returnValues -= returnValues.GetByAncs(exceptionClass);

     // Remove fields with security infrastructure
-    CxList isDeletable3 = (isDeletable2 + isDeletable).GetByAncs(findIfNot);
+    CxList isDel = All.NewCxList(isDeletable2, isDeletable);
+    CxList isDeletable3 = isDel.GetByAncs(findIfNot);
     isDeletable3 -= isDeletable3
-        .GetTargetOfMembers().GetTargetOfMembers().GetTargetOfMembers()
+        .GetTargetOfMembers().GetMembersWithTargets()
         .GetMembersOfTarget().GetMembersOfTarget().GetMembersOfTarget();



-    deleteMembers = returnValues;
+    deleteMembers = All.NewCxList(returnValues);
```

```
    foreach (CxList member in deleteMembers)
    {
        string typeName = member.CxSelectElementValue<CSharpGraph, string>(
@@ -168,18 +173,32 @@
        if (typeName != "")
        {
            CxList methodMember = apexFiles.GetByAncs(apexFiles.GetMethod(member));
-           CxList nonTypeSObjectType = methodMember.FindByMemberAccess("*.sObjectType", false);
+
+           // Handle with member with list as a type
+           if(typeName == "list")
+           {
+               CxList memberDef = decls.FindDefinition(member);
+               CxList memberDefTypeRef = typeReferences.GetByAncs(memberDef) - genericTypeRefs;
+               string memberDefTypeRefName = memberDefTypeRef.GetName();
+
+               // Schema.sObjectType.(memberDefTypeRefName).isDeletable()
+               CxList sObjectType = methodMember.FindByMemberAccess("*sObjectType." + memberDefTypeRefName, false);
+               CxList isDeletableIfStmt = (sObjectType.GetMembersOfTarget() * isDeletableShort).GetAncOfType<IfStmt>();
+               returnValues -= member.GetByAncs(isDeletableIfStmt);
+           }
+
+           CxList nonTypeSobjectType = methodMember.FindByMemberAccess("*.sObjectType", false);
            CxList members = methodMember.FindByMemberAccess(typeName + ".sObjectType", false);
-           nonTypeSObjectType -= members;
-           nonTypeSObjectType.Add(
-               nonTypeSObjectType.GetTargetOfMembers(),
-               nonTypeSObjectType.GetMembersOfTarget()
+           nonTypeSobjectType -= members;
+           nonTypeSobjectType.Add(
+               nonTypeSobjectType.GetTargetOfMembers(),
+               nonTypeSobjectType.GetMembersOfTarget()
                );
            members = members.GetByAncs(findIfNot);
-           CxList methodMembers = members;
+           CxList methodMembers = All.NewCxList(members);
            methodMembers.Add(methodMember);
            //remove references to isDeletable call for other classes
-           methodMembers -= nonTypeSObjectType;
+           methodMembers -= nonTypeSobjectType;
            CxList influenced = methodMembers.DataInfluencingOn(isDeletable) + isDeletable3 * methodMember;
            //check if influenced func is in ignored ifs (with return or throw inside)
            CxList influencedIf = influenced.GetLastNodesInPath().GetAncOfType<IfStmt>();
```

Apex / Apex_Force_com_Serious_Security_Risk / CSRF

Code changes

Apex / Apex_Force_com_Serious_Security_Risk / Dereferenced_Field

Code changes

```
---
+++
@@ -141,6 +141,7 @@
 vfPages.Add(vfPages.GetParameters(methods.FindByShortNames(vfNewElements)));


 result = returnValues * globals;
-result.Add(vfPages.InfluencedByAndNotSanitized(returnValues, sanitize));
+result.Add(vfPages.InfluencedByAndNotSanitized(returnValues, sanitize)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow));


 result -= result.DataInfluencedBy(result);
```

Apex / Apex_Force_com_Serious_Security_Risk / FLS_Create

Code changes

```
---
+++
@@ -3,4 +3,5 @@
   * (https://trailhead.salesforce.com/en/content/learn/modules/data-leak-prevention/prevent-crud-and-fls-violations)
   */
 List<string> operations = new List<string>{ "insert", "convertlead", "undelete", "upsert" };
-result = FLS_Violations(false, "iscreateable", operations);
+result = FLS_Violations(false, "iscreateable", operations)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Apex / Apex_Force_com_Serious_Security_Risk / FLS_Create_Partial

Code changes

```
---
+++
@@ -4,4 +4,5 @@
   */
 List<string> operations = new List<string>{ "insert", "convertlead", "undelete", "upsert" };
 // this will only returns the first vulnerable result by operation
-result = FLS_Violations(true, "iscreateable", operations);
+result = FLS_Violations(true, "iscreateable", operations)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Apex / Apex_Force_com_Serious_Security_Risk / FLS_Read

Code changes

```
---
+++
@@ -1,4 +1,4 @@
-result = FLS_Violations(false, "isAccessible", new List<string> {"select"});
+result = FLS_Violations(false, "isAccessible", new List<string>{"select"});


 result = result.GetFirstNodesInPath();
```

Apex / Apex__Force__com__Serious__Security__Risk / FLS__Update

Code changes

```
---
+++
@@ -3,4 +3,5 @@
  * (https://trailhead.salesforce.com/en/content/learn/modules/data-leak-prevention/prevent-crud-and-fls-violations)
  */
 List<string> operations = new List<string>{ "merge", "update", "upsert", "convertlead" };
-result = FLS_Violations(false, "isupdateable", operations);
+result = FLS_Violations(false, "isupdateable", operations)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Apex / Apex__Force__com__Serious__Security__Risk / FLS__Update__Partial

Code changes

```
---
+++
@@ -4,4 +4,5 @@
  */
 List<string> operations = new List<string>{ "merge", "update", "upsert", "convertlead" };
 // this will only returns the first vulnerable result by operation
-result = FLS_Violations(true, "isupdateable", operations);
+result = FLS_Violations(true, "isupdateable", operations)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Apex / Apex__Force__com__Serious__Security__Risk / Frame__Spoofing

Code changes

```
---
+++
@@ -11,6 +11,7 @@

 sanitize.Add(Find_Id_Sanitizers(),Find_Integers(),testCode);

-result = iframe.InfluencedByAndNotSanitized(inputs, sanitize);
+result = iframe.InfluencedByAndNotSanitized(inputs, sanitize)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

 result -= testCode;
```

Apex / Apex__Force__com__Serious__Security__Risk / inputText__Ignoring__FLS

Code changes

```
---
+++
@@ -32,3 +32,4 @@

 result = inputText.InfluencedByAndNotSanitized(viewMethodParam, Find_Test_Code());
 result.Add(renderedMethod.InfluencedByAndNotSanitized(viewMethodParam, sanitized));
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Apex / Apex_Force_com_Serious_Security_Risk / Insecure_Cookie

Code changes

```
---
+++
@@ -11,9 +11,10 @@

 CxList paramIsSecure = All.GetParameters(cookieClass, 4);

-result = paramIsSecure.DataInfluencedBy(stringFalse);
+result = paramIsSecure.DataInfluencedBy(stringFalse)
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

 CxList isSecure = paramIsSecure.
-    FindByType(typeof(BooleanLiteral)).FindByShortName("false");
+    FindByType<BooleanLiteral>().FindByShortName("false");

 result.Add(isSecure);
```

Apex / Apex_Force_com_Serious_Security_Risk / Insecure_Endpoint

Code changes

```
---
+++
@@ -12,4 +12,4 @@
 CxList httpEndPoint = stringUrlAll.FilterByDomProperty<StringLiteral>(x => x.Value.StartsWith(@"http://")
     && x.Value.Contains(@"http://"));

-result = endPoint.DataInfluencedBy(httpEndPoint);
+result = endPoint.DataInfluencedBy(httpEndPoint).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Apex / Apex_Force_com_Serious_Security_Risk / URL_Redirection_Attack

Code changes

```
---
+++
@@ -28,7 +28,8 @@

 reference -= binary;

-result = inputs.InfluencingOnAndNotSanitized(reference, sanitize);
+result = inputs.InfluencingOnAndNotSanitized(reference, sanitize)
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 result.Add(refInputs);

 result -= result.DataInfluencedBy(result);
```

Apex / Apex_ISV_Quality_Rules / SOQL_Dynamic_null_in_Where

Code changes

```
---
+++
```

```
@@ -57,7 +57,8 @@


            if((varRefInEqualIf.Count == 0 && varRefInNotEqualIf.Count == 0) || //if var not tested
                (varRefInEqualIf.Count > 0 && varToAdd.GetByAncs(varRefInEqualIf).Count > 0) || //if(varToAdd == null) and query inside
-           (varRefInNotEqualIf.Count > 0 && varToAdd.GetByAncs(varRefInNotEqualIf).Count == 0)) { //if(varToAdd == null) and query not inside that if
+           (varRefInNotEqualIf.Count > 0 && varToAdd.GetByAncs(varRefInNotEqualIf).Count == 0)) {
+               //if(varToAdd == null) and query not inside that if
                inVariables.Add(varToAdd);
            }
            match = match.NextMatch();
@@ -68,7 +69,8 @@
 dbQuery.Add(inVariables);


 CxList elementsWillInfluenceDb = unknwnRefs.GetByAncs(allIfsConditions).InfluencedBy(dbQuery).GetLastNodesInPath();
-CxList sanitizerTestElementInIF = unknwnRefs.FindAllReferences(dbQuery).FindByFathers(allIfsConditions) * methods.GetTargetOfMembers();
+CxList sanitizerTestElementInIF = unknwnRefs.FindAllReferences(dbQuery)
+   .FindByFathers(allIfsConditions) * methods.GetTargetOfMembers();
 sanitizerTestElementInIF.Add(elementsWillInfluenceDb * methods.GetTargetOfMembers());


 result = dbQuery.InfluencedByAndNotSanitized(nulls, sanitizerTestElementInIF);
```

Apex / Apex_ISV_Quality_Rules / SOQL_with_All_Fields_in_Loop

Code changes

```
---
+++
@@ -15,4 +15,5 @@
 CxList execQuery = methods.FindByMemberAccess("Database.query", false);
 CxList getMapsForeach = fields.GetFirstNodesInPath();


-result.Add(execQuery.DataInfluencedBy(describe).DataInfluencedBy(getMapsForeach));
+result.Add(execQuery.DataInfluencedBy(describe).DataInfluencedBy(getMapsForeach)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow));
```

Apex / Apex_ISV_Quality_Rules / SOSL_With_Where_Clause

Code changes

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


 CxList searchQuery = methods.FindByMemberAccess("search.query");


-result = searchQuery.DataInfluencedBy(whereSOSL);
+result = searchQuery.DataInfluencedBy(whereSOSL)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);


 result.Add(slStatements * whereSOSL);
```

Apex / Apex_Low_Visibility / Potential_URL_Redirection_Attack

Code changes

```
---
+++
@@ -28,8 +28,9 @@

 reference -= refsBinaries;

-result = inputs.InfluencingOnAndNotSanitized(reference, sanitize);
-result.Add(refInputs);
+result.Add(
+   inputs.InfluencingOnAndNotSanitized(reference, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow),
+   refInputs);

 result -= testCode;
 result -= result.DataInfluencedBy(result);
```

Apex / Apex_Low_Visibility / Use_of_Broken_or_Risky_Cryptographic_Algorithm

Code changes

```
---
+++
@@ -14,4 +14,5 @@

 CxList sanitize = digest.GetTargetOfMembers(); // Eliminate flows throught the target of the digest methods

-result.Add(digest.InfluencedByAndNotSanitized(cryptoFunc, sanitize));
+result.Add(digest.InfluencedByAndNotSanitized(cryptoFunc, sanitize)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow));
```

Apex / Apex_Low_Visibility / Verbose_Error_Reporting

Code changes

```
---
+++
@@ -10,6 +10,7 @@

 CxList testCode = Find_Test_Code();

-result = outputs.InfluencedByAndNotSanitized(exc, testCode);
+result = outputs.InfluencedByAndNotSanitized(exc, testCode)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

 result -= testCode;
```

ASP / ASP_Best_Coding_Practice / Hardcoded_Connection_String

Code changes

```
---
+++
@@ -1,6 +1,6 @@
```

```
 CxList conStr = Find_Strings().FindByName("*provider*", false);


-CxList openConnection = All.FindByType(typeof(ObjectCreateExpr)).FindByShortName("*connection");
+CxList openConnection = Find_ObjectCreations().FindByShortName("*connection");
 CxList openConParam = All.GetParameters(openConnection);


-result.Add(conStr.DataInfluencingOn(openConParam));
+result.Add(conStr.DataInfluencingOn(openConParam).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow));
```

ASP / ASP_Best_Coding_Practice / PersistSecurityInfo_is_True

Code changes

```
---
+++
@@ -1,6 +1,7 @@
 CxList PersistSecurityInfo = All.FindByRegex(@"persist security info(\s)*=(\s)*(true|yes)");


-CxList openConnection = All.FindByType(typeof(ObjectCreateExpr)).FindByShortName("*connection");
+CxList openConnection = Find_ObjectCreations().FindByShortName("*connection");
 CxList openConParam = All.GetParameters(openConnection);


-result = openConParam.DataInfluencedBy(PersistSecurityInfo.FindByType(typeof(StringLiteral)));
+result = openConParam.DataInfluencedBy(PersistSecurityInfo.FindByType<StringLiteral>())
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_High_Risk / Code_Injection

Code changes

```
---
+++
@@ -6,4 +6,4 @@
 CxList code = FindByMemberAccess_ASP("Server.Execute");
 code.Add(executerMethods);


-result = Find_Interactive_Inputs().DataInfluencingOn(code);
+result = Find_Interactive_Inputs().DataInfluencingOn(code).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_High_Risk / Command_Injection

Code changes

```
---
+++
@@ -3,4 +3,4 @@
 CxList inputs = Find_Interactive_Inputs();
 CxList exec = Find_Execute();


-result = exec.DataInfluencedBy(inputs);
+result = exec.DataInfluencedBy(inputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_High_Risk / Connection_String_Injection

Code changes

```
---
+++
@@ -2,4 +2,4 @@

 CxList inputs = Find_Interactive_Inputs();

-result = con.DataInfluencedBy(inputs);
+result = con.DataInfluencedBy(inputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_High_Risk / Dangerous_File_Upload

Code changes

```
---
+++
@@ -3,5 +3,7 @@
 CxList save = All.FindByName("*.postedfile.saveas");
 CxList MapPath = All.FindByName("server.mappath");

-result = save.DataInfluencedBy(file + inputs) -
+result = save.DataInfluencedBy(All.NewCxList(file, inputs)) -
             save.DataInfluencedBy(MapPath);
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_High_Risk / LDAP_Injection

Code changes

```
---
+++
@@ -1,8 +1,10 @@
 CxList inputs = Find_Interactive_Inputs();
 CxList methods = Find_Methods();

-CxList sanitize = Find_Integers() + methods.FindByShortName("replace");
+CxList sanitize = All.NewCxList(Find_Integers(), methods.FindByShortName("replace"));

 CxList de = Find_LDAP();
 result = de.InfluencedByAndNotSanitized(inputs, sanitize);
 result -= result.DataInfluencedBy(result);
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_High_Risk / Resource_Injection

Code changes

```
---
+++
@@ -1,2 +1,3 @@
-CxList socket = All.FindByType(typeof(ObjectCreateExpr)).FindByShortName("tcplistener");
-result = Find_Interactive_Inputs().DataInfluencingOn(All.GetByAncs(socket));
+CxList socket = Find_ObjectCreations().FindByShortName("tcplistener");
```

```
+result = Find_Interactive_Inputs().DataInfluencingOn(All.GetByAncs(socket))
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_High_Risk / XPath_Injection

Code changes

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

 CxList inputs = Find_Interactive_Inputs();
 CxList sanitized = Find_Sanitize();
-result = XPath.InfluencedByAndNotSanitized(inputs, sanitized);
+result = XPath.InfluencedByAndNotSanitized(inputs, sanitized).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_Low_Visibility / Hardcoded_password_in_Connection_String

Code changes

```
---
+++
@@ -5,6 +5,7 @@
 hardcodedPasswords -= hardcodedPasswords.GetParameters(All.FindByMemberAccess("request.form", false));

 // Find hardcoded passwords used in a connection
-CxList sanitizers = Find_Same_Value_Sanitizers_For_Hardcoded_Password(connections + hardcodedPasswords);
+CxList sanitizers = Find_Same_Value_Sanitizers_For_Hardcoded_Password(All.NewCxList(connections, hardcodedPasswords));
-result = connections.InfluencedByAndNotSanitized(hardcodedPasswords, sanitizers);
+result = connections.InfluencedByAndNotSanitized(hardcodedPasswords, sanitizers)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_Low_Visibility / Impersonation_Issue

Code changes

```
---
+++
@@ -3,8 +3,8 @@
 CxList Logon = methods.FindByShortName("logonuser");
 CxList DuplicateToken = methods.FindByShortName("duplicatetoken");

-CxList Impersonate = All.FindByMemberAccess("windowsidentity.impersonate") +
-                     All.FindByShortName("impersonateloggedonuser");
+CxList Impersonate = All.NewCxList(All.FindByMemberAccess("windowsidentity.impersonate"),
+                     All.FindByShortName("impersonateloggedonuser"));

 foreach(CxList curLogon in Logon)
 {
@@ -25,3 +25,5 @@
 }

 result.Add(All.FindByShortName("createprocesswithlogonw").DataInfluencedBy(Inputs));
```

```
+

+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_Low_Visibility / Information_Exposure_Through_an_Error_Message

Code changes

```
---

+++

@@ -1,21 +1,21 @@

-CxList main_decl = (All.FindByType(typeof(MethodDecl)))

+CxList main_decl = Find_MethodDecls()

    .FindByName("*.main").FindByFieldAttributes(Modifiers.Public | Modifiers.Static);


 CxList classes_with_main=All.GetClass(main_decl);


-CxList ctch = All.FindByType(typeof(Catch));

-CxList class_of_ctch = All.GetClass(ctch);

+CxList ctch = Find_Catch();


 CxList class_of_ctch_not_with_main = (All - classes_with_main).GetClass(ctch);


 ctch = ctch.GetByAncs(class_of_ctch_not_with_main);


-CxList class_not_with_main = All.FindByType(typeof(ClassDecl)) - classes_with_main;

+CxList class_not_with_main = Find_ClassDecl() - classes_with_main;

 class_not_with_main = All.GetByAncs(class_not_with_main);


-CxList exc = All.FindAllReferences(ctch) - ctch +

-   All.GetByAncs(class_not_with_main).FindByNames("*server.getlasterror*","*innerexception*");

+CxList exc = All.NewCxList(All.FindAllReferences(ctch) - ctch,

+   All.GetByAncs(class_not_with_main).FindByNames("*server.getlasterror*","*innerexception*"));


 CxList err = All.FindByShortName("err");


-result = Find_Interactive_Outputs().DataInfluencedBy(exc + err);

+result = Find_Interactive_Outputs().DataInfluencedBy(All.NewCxList(exc, err))

+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_Low_Visibility / Insufficiently_Protected_Credentials

Code changes

```
---

+++

@@ -2,6 +2,6 @@


 psw = psw - Find_Methods();


-CxList DB = All.FindByName("*db*") + Find_DB();

+CxList DB = All.NewCxList(All.FindByName("*db*"), Find_DB());
```

```
-result = psw.DataInfluencedBy(DB);
+result = psw.DataInfluencedBy(DB).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_Low_Visibility / Log_Forging

Code changes

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

 CxList sanitize = Find_Integers();

-result = Log.InfluencedByAndNotSanitized(Inputs, sanitize);
+result = Log.InfluencedByAndNotSanitized(Inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_Low_Visibility / Open_Redirect

Code changes

```
---
+++
@@ -1,14 +1,12 @@
-CxList redirect = All.FindByMemberAccess("httpresponse.redirect") +
-   All.FindByName("*response.redirect");
+CxList redirect = All.NewCxList(All.FindByMemberAccess("httpresponse.redirect"),
+   All.FindByName("*response.redirect"));

-CxList inputs = All.FindByMemberAccess("*httprequest.QueryString_*") +
-   All.FindByMemberAccess("*httprequest.QueryString.item") +
-   All.FindByName("*request.Querystring_*") +
-   All.FindByName("*request.QueryString.item") +
-   All.FindByName("*request.item") +
-   All.FindByShortName("request").FindByFathers(All.FindByType(typeof(IndexerRef)));
+CxList inputs = All.NewCxList(
+   All.FindByMemberAccesses(new string[] {"*httprequest.QueryString_*", "*httprequest.QueryString.item"}),
+   All.FindByNames(new string[] {"*request.Querystring_*", "*request.QueryString.item", "*request.item"}),
+   All.FindByShortName("request").FindByFathers(Find_IndexerRefs()));

 inputs -= inputs.DataInfluencingOn(inputs);

 CxList sanitize = Find_Integers();
-result = redirect.InfluencedByAndNotSanitized(inputs, sanitize);
+result = redirect.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_Low_Visibility / Script_Poisoning

Code changes

```
---
+++
@@ -3,4 +3,4 @@
 CxList inputs = Find_Interactive_Inputs();
 CxList timeout = All.FindByName("*Server.ScriptTimeout", false);
```

```
-result = inputs.DataInfluencingOn(timeout);
+result = inputs.DataInfluencingOn(timeout).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_Low_Visibility / Thread_Safety_Issue

Code changes

```
---
+++
@@ -1,13 +1,13 @@
-CxList logs=   All.FindByName("*log*",false)+
-               All.FindByType("logger");
+CxList logs=All.NewCxList(All.FindByName("*log*",false), All.FindByType("logger"));

 CxList no_logs = All-logs;

 CxList statics = no_logs.FindAllReferences(no_logs.FindByFieldAttributes(Modifiers.Static) -
                                            no_logs.FindByFieldAttributes(Modifiers.Readonly));

-statics = statics - no_logs.FindByType(typeof(MethodInvokeExpr));
+statics = statics - no_logs.FindByType<MethodInvokeExpr>();

 CxList EventArgs =  All.FindByType("*commandeventargs");

-result = (EventArgs + Find_Interactive_Inputs()).DataInfluencingOn(statics);
+result = (All.NewCxList(EventArgs,Find_Interactive_Inputs())).DataInfluencingOn(statics)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_Low_Visibility / Trust_Boundary_Violation_in_Session_Variables

Code changes

```
---
+++
@@ -1,6 +1,5 @@
 CxList setAttr = All.FindByMemberAccess("session.add");
-CxList sanitize = Find_Sanitize() +
-   All.FindByShortName("session");
+CxList sanitize = All.NewCxList(Find_Sanitize(), All.FindByShortName("session"));
 CxList input = Find_Inputs();

-result = setAttr.InfluencedByAndNotSanitized(input, sanitize);
+result = setAttr.InfluencedByAndNotSanitized(input, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_Low_Visibility / URL_Canonicalization_Issue

Code changes

```
---
+++
@@ -1,9 +1,9 @@
-CxList inputs =    All.FindByMemberAccess("*httprequest.rawurl") +
-                   All.FindByName("*request.rawurl");
```

```
+CxList inputs = All.NewCxList(All.FindByMemberAccess("*httprequest.rawurl"),
+                   All.FindByName("*request.rawurl"));

-CxList binaryExpr = All.FindByType(typeof(BinaryExpr));
+CxList binaryExpr = Find_BinaryExpr();
 CxList sanitize = All.FindByName("*server.urldecode");

 CxList bin = binaryExpr.InfluencedByAndNotSanitized(inputs, sanitize);

-result = bin.ControlInfluencingOn(All);
+result = bin.ControlInfluencingOn(All).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_Medium_Threat / CSRF

Code changes

```
---
+++
@@ -2,8 +2,6 @@
 CxList db = Find_DB();
 CxList strings = Find_Strings();

-CxList write = strings.FindByName("*update*", StringComparison.OrdinalIgnoreCase) +
-   strings.FindByName("*delete*", StringComparison.OrdinalIgnoreCase) +
-   strings.FindByName("*insert*", StringComparison.OrdinalIgnoreCase);
+CxList write = strings.FindByNames(new string[]{"*update*", "*delete*", "*insert*"}, StringComparison.OrdinalIgnoreCase);
-result = db.DataInfluencedBy(write).DataInfluencedBy(requests);
+result = db.DataInfluencedBy(write).DataInfluencedBy(requests).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_Medium_Threat / DoS_by_Sleep

Code changes

```
---
+++
@@ -1,8 +1,9 @@
 // $ASP*

+CxList methods = Find_Methods();
+
 CxList Inputs = Find_Interactive_Inputs();
-CxList sleep = All.FindByName("WScript.Sleep", false) +                    // vbs
-   All.GetParameters(Find_Methods().FindByShortName("setTimeout"), 1) +    // js
-   All.GetParameters(Find_Methods().FindByShortName("setInterval"), 1);    // js
+CxList sleep = All.NewCxList(All.FindByName("WScript.Sleep", false), // vbs
+   All.GetParameters(methods.FindByShortNames(new string[]{"setTimeout", "setInterval"}), 1));     // js

-result = sleep.DataInfluencedBy(Inputs);
+result = sleep.DataInfluencedBy(Inputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_Medium_Threat / Parameter_Tampering

Code changes

```
---
+++
@@ -3,11 +3,12 @@
 CxList strings = Find_Strings();


 CxList Select = strings.FindByName("*select*", false);
+
 CxList Where = strings.FindByName("*where*", false);
-CxList And = strings.FindByName("*and *", false) +
-   strings.FindByName("* and*", false);


-db  = db.DataInfluencedBy(Select).DataInfluencedBy(Where);
+CxList And = All.NewCxList(strings.FindByNames(new string[]{"*and *", "* and*"}, false));
+
+db = db.DataInfluencedBy(Select).DataInfluencedBy(Where);
 db -= db.DataInfluencedBy(And);


-result = db.DataInfluencedBy(input);
+result = db.DataInfluencedBy(input).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_Medium_Threat / Path_Traversal

Code changes

```
---
+++
@@ -28,4 +28,4 @@
     }
 }


-result = files.InfluencedByAndNotSanitized(Inputs, sanitized);
+result = files.InfluencedByAndNotSanitized(Inputs, sanitized).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_Medium_Threat / SQL_Injection_Evasion_Attack

Code changes

```
---
+++
@@ -4,4 +4,4 @@
 CxList sanitize = Find_SQL_Sanitize();
 CxList db = Find_SQL_DB_In();


-result = db.InfluencedByAndNotSanitized(decode, sanitize);
+result = db.InfluencedByAndNotSanitized(decode, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_Medium_Threat / Untrusted_Activex

Code changes

```
---
+++
```

```
@@ -1,15 +1,12 @@
 // $ASP*

-CxList root = All; // for VBScript client side: AllVbs()

 // VBScript/JavaScript server side: Server.CreateObject (but can go without the "server"
 // JavaScript client side: new ActiveXObject
 // VBScript client side: CreateObject

-CxList activex =
-   root.FindByShortName("CreateObject", false) +       // VBScript
-   root.FindByShortName("ActiveXObject", false);       // JavaScript (untested)
+CxList activex = All.FindByShortNames(new string[] {"CreateObject", "ActiveXObject"}, false);

 CxList inputs = Find_Inputs();

-result = activex.DataInfluencedBy(inputs);
+result = activex.DataInfluencedBy(inputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

ASP / ASP_Medium_Threat / Use_of_Hard_coded_Cryptographic_Key

Code changes

```
---
+++
@@ -3,12 +3,15 @@
 CxList keys = All.FindByShortName("*enc*", false).FindByShortName("*key*", false);

 CxList key_in_lSide = keys.FindByAssignmentSide(CxList.AssignmentSide.Left);
-CxList strLiterals = All.FindByType(typeof(PrimitiveExpr)) - emptyString - NULL;
+CxList strLiterals = Find_PrimitiveExpr() - All.NewCxList(emptyString, NULL);
 CxList lit_in_rSide = strLiterals.FindByAssignmentSide(CxList.AssignmentSide.Right);

 result = key_in_lSide.FindByFathers(key_in_lSide.GetFathers() * lit_in_rSide.GetFathers());
-CxList key_in_Field = All.GetByAncs(All.FindByType(typeof(FieldDecl)) + All.FindByType(typeof(ConstantDecl))) * keys;
+CxList key_in_Field = All.GetByAncs(All.NewCxList(Find_FieldDecls(), Find_ConstantDecl())) * keys;

-CxList sanitizers = Find_Same_Value_Sanitizers_For_Hardcoded_Password(key_in_Field + strLiterals);
+CxList sanitizers = Find_Same_Value_Sanitizers_For_Hardcoded_Password(
+   All.NewCxList(key_in_Field, strLiterals));

 result.Add(key_in_Field.InfluencedByAndNotSanitized(strLiterals, sanitizers));
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Cobol / Cobol_High_Risk / Command_Injection

Code changes

```
---
+++
@@ -5,4 +5,4 @@
```

```
 CxList commandExecution = system.GetByAncs(calls);


-result = inputs.InfluencingOn(commandExecution);
+result = inputs.InfluencingOn(commandExecution).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Cobol / Cobol_High_Risk / Module_Injection

Code changes

```
---
+++
@@ -3,4 +3,4 @@
 CxList callMethods = methodInvokes.FindByShortName("CALL", false);
 CxList parameters = All.GetParameters(callMethods, 0);


-result = inputs.DataInfluencingOn(parameters);
+result = inputs.DataInfluencingOn(parameters).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Cobol / Cobol_High_Risk / Reflected_XSS_All_Clients

Code changes

```
---
+++
@@ -1,4 +1,4 @@
 CxList inputs = Find_Web_Inputs();
 CxList outputs = Find_Web_Outputs();


-result = inputs.InfluencingOn(outputs);
+result = inputs.InfluencingOn(outputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Cobol / Cobol_High_Risk / Resource_Injection

Code changes

```
---
+++
@@ -1,4 +1,4 @@
 CxList inputs = Find_Inputs();
 CxList resources = Find_Resources();


-result = inputs.InfluencingOn(resources);
+result = inputs.InfluencingOn(resources).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Cobol / Cobol_High_Risk / Sql_Injection

Code changes

```
---
+++
@@ -1,4 +1,4 @@
 CxList inputs = Find_Inputs();
 CxList db = Find_DB();


-result = inputs.DataInfluencingOn(db);
```

```
+result = inputs.DataInfluencingOn(db).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Cobol / Cobol_Medium_Threat / Path_Traversal

Code changes

```
---
+++
@@ -1,4 +1,4 @@
 CxList inputs = Find_Inputs();
 CxList read = Find_Read();

-result = inputs.InfluencingOn(read);
+result = inputs.InfluencingOn(read).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CPP / CPP_Buffer_Overflow / Buffer_Improper_Index_Access

Code changes

```
---
+++
@@ -3,16 +3,14 @@
 CxList ints = Find_Integer_Literals();
 CxList inputs = Find_Inputs();
 CxList methods = Find_Methods();
-CxList conditions = Find_Conditions();
 CxList indexAccess = Find_IndexerRefs();
 CxList paramDeclarators = Find_ParamDecl();
 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
+//General Improper Index Access Usage
 CxList nodes = Find_Improper_Index_Access(false);
 //Creates flow with indexerRefs that are influenced by user input
 CxList unkRefsIndexer = unkRefs.FindByFathers(nodes);
@@ -46,103 +44,3 @@
 }

 result = nodes;
-
-/* 2. Accessing input string by length without sanitizing
-   Sample code
-
-   if (fgets(buf, sizeof(buf), fptr)) {
-     buf[strlen(buf) - 1] = '\0';  // diagnostic required
-     return puts(buf);
-   }                                                    */
-
```

```
-// 2.1 Inputs
-// Get all unknown references that are used as parameters of strlen
-
-CxList strings = unkRefs.FindAllReferences(unkRefs.GetParameters(strlenCalls, 0));
-// Get those that are paramDecl and inputs
-CxList stringParameters = strings.GetAncOfType<ParamDecl>() * inputs;
-// Add those that are used in input methods
-stringParameters.Add(strings.GetByAncs(inputs));
-// Now find all references to the relevant variables
-stringParameters = unkRefs.FindAllReferences(stringParameters);
-// But remove if they are part of sizeof
-stringParameters -= stringParameters.GetByAncs(methods.FindByShortNames("sizeof"));
-
-// 2.2 Sinks
-// Gets problematic accesses
-CxList binOps = binExpr.FilterByDomProperty<BinaryExpr>(
-   x => x.Operator == BinaryOperator.Add || x.Operator == BinaryOperator.Subtract);
-// Grab strlens inside indexRefs
-CxList strlens = methods.FindByShortNames("strlen").GetByAncs(binOps).FindByParameters(stringParameters);
-CxList indexRefs = strlens.GetAncOfType<IndexerRef>().FindAllReferences(stringParameters);
-
-// 2.3 Sanitizers
-// Conditions that are not part of the inputs
-CxList sanitizers = stringParameters.GetByAncs(conditions);
-sanitizers -= stringParameters.GetByAncs(inputs);
-
-// 2.4 Add paths
-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));
-
-CxList assignSanitizers = All.NewCxList();
-// Relevant Binary Operators
-CxList sanitizerBinOps = binExpr.FilterByDomProperty<BinaryExpr>(a => a.Operator != BinaryOperator.Subtract &&
-   a.Operator != BinaryOperator.Add && a.Operator != BinaryOperator.Divide && a.Operator != BinaryOperator.Multiply);
-// Assignment using the relevant operators
-assignSanitizers.Add(
-   Find_AssignExpr().FilterByDomProperty<AssignExpr>(a => a.Operator != AssignOperator.Assign),
-   sanitizerBinOps.GetAncOfType<AssignExpr>());
```

```
-// UnkRefs used in the left side of assignments (or declarators)
-CxList assignSanitizersUnkRefs = All.NewCxList();
-assignSanitizersUnkRefs.Add(
-   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*")
-   );
-
-// get only indices influenced by inputs and not sanitized
-indices = indices.InfluencedByAndNotSanitized(inputs, idxSanitizers).GetLastNodesInPath();
-// paths from inputs to conditions
-CxList inputToCondition = inputs.DataInfluencingOn(conditions);
-
-// those referencess that are used in a condition
-CxList unkRefIndices = unkRefs.GetByAncs(inputToCondition.GetLastNodesInPath());
-
-// remove those indices that were used in a condition
-indices -= indices.GetByAncs(inputToCondition.GetLastNodesInPath().GetFathers());
-
-foreach (CxList indice in indices) {
-   CxList currIfVar = unkRefs.FindAllReferences(indice) * unkRefIndices;
-
-   if (currIfVar.Count > 0) {
-       CSharpGraph ifVarGraph = currIfVar.GetFirstGraph();
-       CSharpGraph divVarGraph = indice.GetFirstGraph();
-
-       if (ifVarGraph.ShortName == divVarGraph.ShortName &&
-           ifVarGraph != divVarGraph &&
-           ifVarGraph.LinePragma.FileName == divVarGraph.LinePragma.FileName &&
-           ifVarGraph.LinePragma.Line < divVarGraph.LinePragma.Line)
-       {
-           indices -= indice;
-       }
-   }
-}
-
-result.Add(indices);
```

CPP / CPP_Buffer_Overflow / Format_String_Attack

Code changes

```
---
+++
```

```
@@ -1,26 +1,26 @@
-CxList inputs = Find_Interactive_Inputs();
-inputs.Add(Find_Read());
-inputs.Add(Find_DB());
-
+CxList inputs = All.NewCxList(Find_Interactive_Inputs(),Find_Read(),Find_DB());
 CxList format = Get_Format_Parameter();
-CxList sanitize = Find_Integers();
+CxList sanitize = All.NewCxList(Find_Integers(),format.FindByType<StringLiteral>());

 result = inputs.InfluencingOnAndNotSanitized(format, sanitize);

 // Now find all cases in which the number of expected args is bigger then the number of actual args
-var StringLiterals = Find_Strings();
+CxList StringLiterals = Find_Strings();
 format = format * StringLiterals;
+// Create Used Regex's
+Regex rgxPercent = new Regex("%");
+Regex rgxPercentNum = new Regex("%\\d*\\$");
+Regex rgxDoublePercent = new Regex("%%");
 foreach(CxList formatString in format){
-   var formatMethods = formatString.GetAncOfType(typeof(MethodInvokeExpr));
-   var numOfParams = (All.GetParameters(formatMethods)).Count / 2;
-   var formatedStringPosition = formatString.GetFathers().GetIndexOfParameter();
-   var formatStringName = formatString.GetName();
-   var percentagesInFormatString = Regex.Matches(formatStringName, "%").Count;
-   percentagesInFormatString -= Regex.Matches(formatStringName, "%\\d+?\\$").Count; // Remove %num$
+   CxList formatMethods = formatString.GetAncOfType<MethodInvokeExpr>();
+   int numOfParams = (All.GetParameters(formatMethods)).Count / 2;
+   int formatedStringPosition = formatString.GetFathers().GetIndexOfParameter();
+   string formatStringName = formatString.GetName();
+   int percentagesInFormatString = rgxPercent.Matches(formatStringName).Count;
+   percentagesInFormatString -= rgxPercentNum.Matches(formatStringName).Count; // Remove %num$
    while (formatStringName.Contains("%%")){ // Remove %%
        percentagesInFormatString -= 2;
-       var rgx = new Regex("%%");
-       formatStringName = rgx.Replace(formatStringName, "", 1);
+       formatStringName = rgxDoublePercent.Replace(formatStringName, "", 1);
    }

    if (percentagesInFormatString > numOfParams - 1 - formatedStringPosition)
```

CPP / CPP_Low_Visibility / NULL_Pointer_Dereference

Code changes

```
---
+++
@@ -6,10 +6,15 @@
 CxList unknownReferences = Find_Unknown_References();
```

```
 CxList conditions = Find_Conditions();
 CxList ifs = Find_Ifs();
-CxList pointerTotarget = pointers.GetByAncs(All.NewCxList(unary.FindByShortName("Pointer"),
-                                                  pointers.FindByType<IndexerRef>()));
 CxList equalsNullCond = Find_Ifs_NullCondition(true);
 CxList notEqualsNullCond = Find_Ifs_NullCondition(false);
+CxList pointerUnary = unary.FindByShortName("Pointer");
+CxList pointerTotarget = pointers.GetByAncs(All.NewCxList(pointerUnary, pointers.FindByType<IndexerRef>()));
+
+// Remove pointers used as IndexerRef Indices (Do not remove cases where these pointers are accessed)
+CxList pointersInIndices = pointerTotarget.GetByAncs(Find_IndexerRefs().CxSelectElements<IndexerRef>(x => x.Indices, -1));
+pointersInIndices -= pointersInIndices.FindByFathers(pointerUnary);
+pointerTotarget -= pointersInIndices;

 // Remove pointers that check pointer condition
 CxList pointerCondition = pointers.GetByAncs(conditions);
@@ -115,7 +120,6 @@

 // Sanitize &*
 CxList address = unary.FindByShortName("Address");
-CxList pointerUnary = unary.FindByShortName("Pointer");
 CxList addressChild = pointerUnary.FindByFathers(address);
 sanitizers.Add(pointers.FindByFathers(addressChild));

@@ -157,7 +161,10 @@
 // Clear flows where the dereference originated directly from the right hand side.
 // (Ie: 'arr' in arr[i] = elem, when elem == NULL)
 CxList lastNode = pointerToInfluencing.GetLastNodesInPath();
-CxList lastAssigners = All.NewCxList(lastNode.GetAssigner(), lastNode.GetFathers().GetAssigner());
+CxList lastNodeFather = lastNode.GetFathers();
+CxList lastAssigners = All.NewCxList(lastNode.GetAssigner(),
+                                     lastNodeFather.GetAssigner(),
+                                     lastNodeFather.GetMembersOfTarget().GetAssigner());
 // Remove Impossible flows, where an allegdly null pointer completes a Member Access Operation before dereference.
 CxList refsWithMembers = unknownReferences.GetMembersOfTarget().GetTargetOfMembers();
 refsWithMembers -= lastNode;
```

CPP / CPP_Low_Visibility / Unchecked_Array_Index

Code changes

```
---
+++
@@ -1,10 +1,10 @@
 // Helper delegate to compare two expression (allocated array size with index access)
-Func<CxList, CxList, bool> CompareArrayWithIndex = delegate(CxList arrayRef, CxList index)
-{
-   Func<Expression, IntegerIntervalAbstractValue> GetExprAbsValue = delegate(Expression expression)
-   {
+Func < CxList, CxList, bool> CompareArrayWithIndex = delegate(CxList arrayRef, CxList index)
```

```
+   {
+   Func < Expression, IntegerIntervalAbstractValue > GetExprAbsValue = delegate(Expression expression)
+       {
        IntegerIntervalAbstractValue expressionAbsValue = null;
-       if(expression != null)
+       if (expression != null)
        {
            IAbstractValue absValue = expression.AbsValue;
            if (absValue is ObjectAbstractValue objectAbsValue)
@@ -17,24 +17,25 @@
            }
        }
        return expressionAbsValue;
-   };
-
+       };
+
    Expression arrayRefExpr = arrayRef.TryGetCSharpGraph<Expression>();
    Expression indexExpr = index.TryGetCSharpGraph<Expression>();
    IntegerIntervalAbstractValue arrayAbsValue = GetExprAbsValue(arrayRefExpr);
    IntegerIntervalAbstractValue indexAbsValue = GetExprAbsValue(indexExpr);
-
+
    if (arrayAbsValue != null && indexAbsValue != null)
    {
        IAbstractValue isValidIndex = indexAbsValue.LessThan(arrayAbsValue);
        return (isValidIndex is TrueAbstractValue);
    }
    return false;
-};
-
-// Find all variables in indexes
+   };
+
+//Find all variables in indexes
 CxList indexerRefs = Find_IndexerRefs();
-CxList possibleIndexes = Find_UnknownReference().GetByAncs(indexerRefs);
+CxList unkRefs = Find_UnknownReference();
+CxList possibleIndexes = unkRefs.GetByAncs(indexerRefs);
 possibleIndexes -= possibleIndexes.FindByType("string");
 CxList indexes = All.NewCxList();
 bool isValidIndex;
@@ -43,14 +44,14 @@
    isValidIndex = false;
    CxList indiceTarget = indexerRef.CxSelectDomProperty<IndexerRef>(x => x.Target);
    CxList indices = indexerRef.CxSelectElements<IndexerRef>(i => i.Indices);
-
-   if(indices.Count == 1)
```

```
+
+   if (indices.Count == 1)
    {
        // use absint to compare array alocated size with index access
        isValidIndex = CompareArrayWithIndex(indiceTarget, indices);
    }
-
-   if(!isValidIndex)
+
+   if (!isValidIndex)
    {
        indexes.Add(indices * possibleIndexes);
    }
@@ -67,5 +68,148 @@
    goodIndexes.Add(indexes.FindAllReferences(checkedReference).GetByAncs(condition));
 }
 CxList problems = indexes - goodIndexes;
-CxList problemsLefAssign = problems.GetAncOfType<IndexerRef>().FindByAssignmentSide(CxList.AssignmentSide.Left);
+CxList problemsLefAssign = problems
+   .GetAncOfType<IndexerRef>()
+   .FindByAssignmentSide(CxList.AssignmentSide.Left);
 result = problems.GetByAncs(problemsLefAssign);
+
+// Moved from Buffer_Improper_Index_Access
+
+/* 2. Accessing input string by length without sanitizing
+  Sample code
+
+  if (fgets(buf, sizeof(buf), fptr)) {
+ buf[strlen(buf) - 1] = '\0';  // diagnostic required
+ return puts(buf);
+  }                                                    */
+
+// 2.1 Inputs
+// Get all unknown references that are used as parameters of strlen
+CxList conditions = Find_Conditions();
+CxList binExpr = Find_BinaryExpr();
+CxList inputs = Find_Inputs();
+CxList methods = Find_Methods();
+CxList paramDeclarators = Find_ParamDecl();
+CxList allDeclarators = Find_Declarators();
+allDeclarators.Add(paramDeclarators);
+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;
```

```
+
+// Add those that are used in input methods
+stringParameters.Add(strings.GetByAncs(inputs));
+
+// Now find all references to the relevant variables
+stringParameters = unkRefs.FindAllReferences(stringParameters);
+
+// But remove if they are part of sizeof
+stringParameters -= stringParameters.GetByAncs(methods.FindByShortNames("sizeof"));
+
+// 2.2 Sinks
+// Gets problematic accesses
+CxList binOps = binExpr.FilterByDomProperty<BinaryExpr>(x =>
+   x.Operator == BinaryOperator.Add || x.Operator == BinaryOperator.Subtract
+   );
+
+// Grab strlens inside indexRefs
+CxList strlens = methods
+   .FindByShortNames("strlen")
+   .GetByAncs(binOps)
+   .FindByParameters(stringParameters);
+CxList indexRefs = strlens.GetAncOfType<IndexerRef>().FindAllReferences(stringParameters);
+
+// 2.3 Sanitizers
+// Conditions that are not part of the inputs
+CxList sanitizers = stringParameters.GetByAncs(conditions);
+sanitizers -= stringParameters.GetByAncs(inputs);
+
+// 2.4 Add paths
+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 arrayIndices = indexerRefs
+   .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)
+   );
```

```
+
+CxList assignSanitizers = All.NewCxList();
+
+// Relevant Binary Operators
+CxList sanitizerBinOps = binExpr.FilterByDomProperty<BinaryExpr>(a =>
+   a.Operator != BinaryOperator.Subtract
+   && a.Operator != BinaryOperator.Add
+   && a.Operator != BinaryOperator.Divide
+   && a.Operator != BinaryOperator.Multiply
+   );
+
+// Assignment using the relevant operators
+assignSanitizers.Add(
+   Find_AssignExpr().FilterByDomProperty<AssignExpr>(a => a.Operator != AssignOperator.Assign),
+   sanitizerBinOps.GetAncOfType<AssignExpr>()
+   );
+
+// UnkRefs used in the left side of assignments (or declarators)
+CxList assignSanitizersUnkRefs = All.NewCxList();
+assignSanitizersUnkRefs.Add(
+   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*")
+   );
+
+// get only indices influenced by inputs and not sanitized
+arrayIndices = arrayIndices.InfluencedByAndNotSanitized(inputs, idxSanitizers).GetLastNodesInPath();
+
+// paths from inputs to conditions
+CxList inputToCondition = inputs.DataInfluencingOn(conditions);
+
+// those referencess that are used in a condition
+CxList unkRefIndices = unkRefs.GetByAncs(inputToCondition.GetLastNodesInPath());
+
+// remove those indices that were used in a condition
+arrayIndices -= arrayIndices.GetByAncs(inputToCondition.GetLastNodesInPath().GetFathers());
+
+foreach (CxList indice in arrayIndices)
+{
+   CxList currIfVar = unkRefs.FindAllReferences(indice) * unkRefIndices;
```

```
+
+   if (currIfVar.Count > 0)
+   {
+       CSharpGraph ifVarGraph = currIfVar.GetFirstGraph();
+       CSharpGraph divVarGraph = indice.GetFirstGraph();
+
+       if (
+           ifVarGraph.ShortName == divVarGraph.ShortName
+           && ifVarGraph != divVarGraph
+           && ifVarGraph.LinePragma.FileName == divVarGraph.LinePragma.FileName
+           && ifVarGraph.LinePragma.Line < divVarGraph.LinePragma.Line
+           )
+       {
+           arrayIndices -= indice;
+       }
+   }
+}
+
+result.Add(arrayIndices);
```

CPP / CPP_Medium_Threat / Divide_By_Zero

Code changes

```
---
+++
@@ -117,7 +117,7 @@
    CxList currIfVar = unknown.FindAllReferences(varRef) * ifVars;

    if (currIfVar.Count == 0 && varRef.GetByAncs(iterations).Count > 0)
-       currIfVar = unknown.FindAllReferences(varRef) * relevBreakIfs;
+           currIfVar = unknown.FindAllReferences(varRef) * relevBreakIfs;

    if (currIfVar.Count > 0) {

@@ -148,7 +148,7 @@
 result.Add(zeroAbsVal);


-////////////////////////////
+//-------------------------
 //Numeric variables whose abstract value is AnyAbstractValues may contain 0, so they should be part of the result
 CxList decimals = Find_Integers() * unknown;
 decimals.Add(unknown.FindByTypes("float","double"));
@@ -174,12 +174,12 @@
 //Find all possibleZeros that are initialized with a specific real value different from 0
 CxList varsInitWithNoZero = All.FindDefinition(possibleZeros).FindByAssignmentSide(CxList.AssignmentSide.Left);
 CxList allRealLiterals = Find_RealLiterals() - zero;
-varsInitWithNoZero = allRealLiterals.GetAssignee();
+varsInitWithNoZero.Add(allRealLiterals.GetAssignee());
```

```
 possibleZerosToRemove.Add(possibleZeros.FindAllReferences(varsInitWithNoZero));

 possibleZeros -= possibleZerosToRemove;
 result.Add(rightToDiv * possibleZeros);
-//////////////////////-
+//--------------------

 // Add real literal zeros (0.0) on the right side of divide operations to the results.
 CxList literalZeros = rightToDiv * zero;
```

CPP / CPP_Medium_Threat / Pointer_Subtraction_Determines_Size

Code changes

```
---
+++
@@ -118,6 +118,7 @@
    unknownRef.FindAllReferences(Find_Array_Declaration().GetAncOfType<Declarator>()));
 pointers -= pointers.GetByAncs(Find_StructDecl());
 pointers -= pointers.FindByFathers(Find_MemberAccesses());
+pointers -= Find_ReferencePointers();

 // find any pointer or cast to pointer affecting the left/right of a binary subtraction
 CxList binaries = Find_BinaryExpr();
```

CSharp / CSharp_Best_Coding_Practice / Hardcoded_Connection_String

Code changes

```
---
+++
@@ -3,4 +3,4 @@
 CxList openConnection = Find_ObjectCreations().FindByShortName("*Connection");
 CxList openConParam = Find_Param().CxSelectDomProperty<Param>(p => p.Value).GetParameters(openConnection);

-result = conStr.DataInfluencingOn(openConParam);
+result = conStr.DataInfluencingOn(openConParam).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Best_Coding_Practice / PersistSecurityInfo_is_True

Code changes

```
---
+++
@@ -3,4 +3,5 @@
 CxList openConnection = Find_ObjectCreations().FindByShortName("*Connection");
 CxList openConParam = Find_Param().CxSelectDomProperty<Param>(p => p.Value).GetParameters(openConnection);

-result = openConParam.DataInfluencedBy(PersistSecurityInfo.FindByType<StringLiteral>());
+result = openConParam.DataInfluencedBy(PersistSecurityInfo.FindByType<StringLiteral>())
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_High_Risk / Deserialization_of_Untrusted_Data_MSMQ

Code changes

```
---
+++
@@ -1,23 +1,36 @@
 CxList methods = Find_Methods();
 CxList memberAccesses = Find_MemberAccesses();
+CxList paramValue = Find_Param().CxSelectDomProperty<Param>(p => p.Value);

 CxList MSMQInput = Find_Deserialization_Inputs_MSMQ();
 CxList binaryMessageFormatter = All.FindByType("BinaryMessageFormatter");
-CxList relevantBinaryFormatter = MSMQInput.InfluencedBy(binaryMessageFormatter).GetFirstNodesInPath();
+CxList relevantBinaryMessageFormatter = MSMQInput.InfluencedBy(binaryMessageFormatter).GetFirstNodesInPath();

 CxList messageBody = memberAccesses.FindByMemberAccesses(new []{"Message.Body","Message.BodyStream"});
-result.Add(messageBody.InfluencedBy(relevantBinaryFormatter).ReduceFlow(CxList.ReduceFlowType.ReduceSmallFlow));

 CxList binaryFormatterRead = methods.FindByMemberAccess("BinaryMessageFormatter.Read");
-result.Add(binaryFormatterRead.InfluencedBy(MSMQInput));

 // It is unsafe unless a safe SerializationBinder implementation is used, that validates the type to be deserialized
-
 CxList deserialization = methods.FindByMemberAccess("BinaryFormatter.Deserialize");

 // Find all deserialize methods and subtract the ones influenced by good SerializationBinders
 CxList serializationBinders = Find_SerializationBinder_References();
+serializationBinders.Add(serializationBinders.GetFathers());
 CxList unsafeSerializationBinders = Find_Unsafe_Implementation_of_SerializationBinder();
-deserialization -= deserialization.InfluencedByAndNotSanitized(serializationBinders, unsafeSerializationBinders);

-result.Add(MSMQInput.InfluencingOn(deserialization));
+CxList deserializeToRemove = deserialization.InfluencedByAndNotSanitized(serializationBinders,
+   unsafeSerializationBinders).GetLastNodesInPath();
+deserialization -= deserializeToRemove;
+
+result.Add(
+   messageBody.InfluencedBy(relevantBinaryMessageFormatter),
+   binaryFormatterRead.InfluencedBy(MSMQInput),
+   MSMQInput.InfluencingOn(deserialization));
+
+CxList lastNodeValues = result.GetLastNodesInPath();
+
+CxList safeResults = lastNodeValues * paramValue.GetParameters(deserializeToRemove);
+safeResults.Add(lastNodeValues.GetByAncs(methods.FindByMemberAccess("Console.WriteLine").GetFathers()));
+result -= result.IntersectWithNodes(safeResults);
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_High_Risk / Reflected_XSS_All_Clients

Code changes

```
---
+++
@@ -2,14 +2,15 @@
 {
     CxList methods = Find_Methods();
     CxList parameters = Find_Parameters();
+    CxList memberAccesses = Find_MemberAccesses();
+    CxList interactiveInputs = Find_Interactive_Inputs();

-    CxList inputs = All.NewCxList(Find_Interactive_Inputs(), Find_Cloud_Interactive_Inputs());
+    CxList inputs = All.NewCxList(interactiveInputs, Find_Cloud_Interactive_Inputs());
     inputs -= Find_ListBox_Safe_Inputs();

     CxList outputs = Find_XSS_Outputs() - Find_Console_Outputs();

     CxList sanitized = Find_XSS_Sanitize();
-
     //Add LINQ with HttpUtility.HtmlEncode as sanitizer
     CxList linqWhereAndSelect = methods.FindByShortNames(new [] {"where", "select"}, false);
     CxList lambdaMethods = methods.GetByAncs(Find_LambdaExpr());
@@ -26,7 +27,54 @@
     // Adding Minimal API Filters' related sanitizers
     sanitized.Add(Find_ASP_MVC_Minimal_Filter_Sanitizers(sanitized));

-    result = inputs.InfluencingOnAndNotSanitized(outputs, sanitized).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
+    // SetInnerText can be a sanitizer
+    sanitized.Add(methods.FindByMemberAccess("TagBuilder.SetInnerText").GetTargetOfMembers());
+
+    result = outputs.InfluencedByAndNotSanitized(inputs, sanitized).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
+
+    // Filter Attributes to match the vulnerable list<string> below.
+    // Remove any results that don't match the list.
+    List<string> tagAttributes = new List<string> {
+        //Attributes
+        "src", "href", "data", "srcdoc", "action", "target", "value", "content",
+        //Events
+        "onafterprint", "onafterscriptexecute", "onanimationcancel","onanimationend",
+        "onanimationiteration", "onanimationstart", "onauxclick", "onbeforecopy",
+        "onbeforecut", "onbeforeinput", "onbeforeprint", "onbeforescriptexecute",
+        "onbeforetoggle", "onbeforeunload", "onbegin", "onblur", "onbounce", "oncanplay",
+        "oncanplaythrough", "onchange", "onclick",  "onclose", "oncontextmenu", "oncopy",
+        "oncuechange", "oncut", "ondblclick", "ondrag", "ondragend", "ondragenter",
+        "ondragexit", "ondragleave", "ondragover", "ondragstart", "ondrop",
+        "ondurationchange", "onend", "onended", "onerror", "onfinish", "onfocus", "onfocusin",
+        "onfocusout", "onformdata", "onfullscreenchange", "onhashchange", "oninput", "oninvalid",
+        "onkeydown", "onkeypress", "onkeyup", "onload", "onloadeddata", "onloadedmetadata",
+        "onloadstart", "onmessage", "onmousedown", "onmouseenter", "onmouseleave", "onmousemove",
+        "onmouseout", "onmouseover", "onmouseup", "onmousewheel", "onmozfullscreenchange",
```

```
+       "onpagehide", "onpageshow", "onpaste", "onpause", "onplay", "onplaying", "onpointerdown",
+       "onpointerenter", "onpointerleave", "onpointermove", "onpointerout", "onpointerover",
+       "onpointerrawupdate", "onpointerup", "onpopstate", "onprogress", "onratechange", "onrepeat",
+       "onreset", "onresize", "onscroll", "onscrollend", "onsearch", "onseeked", "onseeking",
+       "onselect", "onselectionchange", "onselectstart", "onshow", "onstart", "onsubmit",
+       "onsuspend", "ontimeupdate", "ontoggle", "ontouchend", "ontouchmove", "ontouchstart",
+       "ontransitioncancel", "ontransitionend", "ontransitionrun", "ontransitionstart",
+       "onunhandledrejection", "onunload", "onvolumechange", "onwebkitanimationend",
+       "onwebkitanimationiteration", "onwebkitanimationstart", "onwebkittransitionend", "onwheel",
+       //Events for bootstrap
+       "onanimationstart", "ontransitionend"
+       };
+
+   CxList tagBuilderAttributes = memberAccesses.FindByMemberAccess("TagBuilder.Attributes");
+   CxList indicesIndexerRef = tagBuilderAttributes.GetAncOfType<IndexerRef>()
+       .CxSelectDomProperty<IndexerRef>(x => x.Indices[0]);
+
+   CxList vulnerableAttributes = indicesIndexerRef
+       .FindByAbstractValues(Find_Strings().FindByShortNames(tagAttributes))
+       .GetFathers();
+   vulnerableAttributes.Add(indicesIndexerRef.InfluencedBy(interactiveInputs).GetLastNodesInPath().GetFathers());
+
+   CxList vulnerableAttributesMemberAccess = memberAccesses.FindDescendantsOfType<MemberAccess>(vulnerableAttributes);
+
+   result -= result.IntersectWithNodes(tagBuilderAttributes - vulnerableAttributesMemberAccess);


    // When the flow goes through an "Remote input" (a request to a remote server) it is not vulnerable
    result -= result.IntersectWithNodes(Find_Remote_Requests());
```

CSharp / CSharp_Low_Visibility / Heap_Inspection

Code changes

```
---
+++
@@ -8,6 +8,8 @@
 CxList propertyList = Find_PropertyDecl();
 CxList arrayCreateList = Find_ArrayCreateExpr();
 CxList statements = Find_Statements();
+CxList methodDecls = Find_MethodDecls();
+CxList returnStmts = Find_ReturnStmt();

 /*
    INPUTS - Passwords - string and char[]
@@ -85,4 +87,14 @@
 CxList assignVarDeclarators = declaratorsList.FindDefinition(assignVar);
 passwordsValid -= assignVarDeclarators.GetAncOfType<VariableDeclStmt>();

+//Password Hashes / BrowserKeys / Additional encrypted
+CxList encryptMethods = methodsList.FindByShortNames(methodDecls.GetMethod(encrypt.FindByFathers(returnStmts)).GetName());
```

```
+
+CxList toRemove = All.NewCxList(
+   passwordsValid.FindByShortName("passwordHash", false),
+   Find_MemberAccesses().FindByMemberAccesses("AccessKeys", new []{"BrowserOperaAccessKey", "BrowserAccessKey"})
+   .GetAssignee() * passwordsValid,
+   declaratorsList.FindAllReferences(encryptMethods.GetAssignee()) * passwordsValid);
+passwordsValid -= toRemove;
+
 result = passwordsValid;
```

CSharp / CSharp_Low_Visibility / Impersonation_Issue

Code changes

```
---
+++
@@ -42,3 +42,4 @@
 }
 result.Add(Impersonate.DataInfluencedBy(prms));
 result.Add(All.FindByShortName("CreateProcessWithLogonW").DataInfluencedBy(Inputs));
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Low_Visibility / Information_Leak_Through_Persistent_Cookies

Code changes

```
---
+++
@@ -3,4 +3,4 @@
 CxList cookie = All.FindByName("*Response.Cookies*", false);
 cookie.Add(All.FindByMemberAccess("HttpResponse.Cookies*"));

-result = cookie.InfluencedBy(psw);
+result = cookie.InfluencedBy(psw).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Low_Visibility / Leaving_Temporary_Files

Code changes

```
---
+++
@@ -9,3 +9,5 @@
         result.Add(curTmpFile);
     }
 }
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Low_Visibility / Open_Redirect

Code changes

```
---
+++
@@ -36,3 +36,5 @@
     .GetByAncs(aspNetControllers);
```

```
 result.Add(controllerRedirects.InfluencedByAndNotSanitized(Find_Interactive_Inputs(), sanitize));
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Low_Visibility / Reliance_on_DNS_Lookups_in_a_Decision

Code changes

```
---
+++
@@ -7,4 +7,4 @@
 CxList DNS = IPHostEntry.InfluencedBy(hostIPAddress);
 DNS.Add(IPHostEntryName);

-result = cond.InfluencedBy(DNS);
+result = cond.InfluencedBy(DNS).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Low_Visibility / Stored_Command_Argument_Injection

Code changes

```
---
+++
@@ -20,4 +20,4 @@

 exec.Add(processWriteLines);

-result = exec.InfluencedByAndNotSanitized(inputs, sanitize);
+result = exec.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Low_Visibility / Thread_Safety_Issue

Code changes

```
---
+++
@@ -38,5 +38,5 @@
     CxList sanitizers = All.FindByFathers(returns.GetByAncs(parameters));

     CxList source = All.NewCxList(EventArgs, inputs);
-    result = source.InfluencingOnAndNotSanitized(statics, sanitizers);
+    result = source.InfluencingOnAndNotSanitized(statics, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);;
 }
```

CSharp / CSharp_Low_Visibility / Trust_Boundary_Violation_in_Session_Variables

Code changes

```
---
+++
@@ -36,7 +36,7 @@
 sinks.Add(memberAccesses.FindByShortNames(new [] {"Session_*", "Session"})
     .FindByFathers(indexerRefs.FindByAssignmentSide(CxList.AssignmentSide.Left)));

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

```
+result = sinks.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);


 // When the flow goes through an "Remote input" (a request to a remote server) it is not vulnerable
 result -= result.IntersectWithNodes(Find_Remote_Requests());
```

CSharp / CSharp_Low_Visibility / URL_Canonicalization_Issue

Code changes

```
---
+++
@@ -4,6 +4,6 @@
 CxList binaryExpr = Find_BinaryExpr();
 CxList sanitize = All.FindByName("*Server.UrlDecode");

-CxList bin = binaryExpr.InfluencedByAndNotSanitized(inputs, sanitize);
+CxList bin = binaryExpr.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

 result = bin.ControlInfluencingOn(All);
```

CSharp / CSharp_Low_Visibility / Use_of_RSA_Algorithm_without_OAEP

Code changes

```
---
+++
@@ -6,3 +6,5 @@
 result = All.NewCxList(
     rsaParam * negative,
     rsaParam.DataInfluencedBy(negative));
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Medium_Threat / CGI_XSS

Code changes

```
---
+++
@@ -54,4 +54,5 @@
             result.Add(output);
         }
     }
+    result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

CSharp / CSharp_Medium_Threat / Data_Filter_Injection

Code changes

```
---
+++
@@ -4,4 +4,4 @@
 CxList inputs = All.NewCxList(Find_Interactive_Inputs(), Find_Cloud_Interactive_Inputs());
 CxList sanitized = Find_SQL_Sanitize();
```

```
-result = inputs.InfluencingOnAndNotSanitized(db, sanitized);
+result = inputs.InfluencingOnAndNotSanitized(db, sanitized).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Medium_Threat / DoS_by_Sleep

Code changes

```
---
+++
@@ -35,3 +35,5 @@

 result = concatPaths;
 result.Add(resultSleep);
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Medium_Threat / Hardcoded_password_in_Connection_String

Code changes

```
---
+++
@@ -34,6 +34,6 @@
 CxList possibleMethodsAndPsw = All.NewCxList(possibleMethods, psw);
 sanitizers = Find_Same_Value_Sanitizers_For_Hardcoded_Password(possibleMethodsAndPsw);

-result.Add(possibleMethods.InfluencedByAndNotSanitized(psw, sanitizers));
+result.Add(possibleMethods.InfluencedByAndNotSanitized(psw, sanitizers), connectionStrings);
-result.Add(connectionStrings);
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Medium_Threat / Improper_Restriction_of_XXE_Ref

Code changes

```
---
+++
@@ -32,4 +32,4 @@
 CxList vulnerableXmlReaderXXE = xmlReaderResults.IntersectWithNodes(xmlReaderXxeInfluencedByParse);
 XXE.Add(vulnerableXmlReaderXXE);

-result = XXE.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = XXE.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Medium_Threat / Insecure_Cookie

Code changes

```
---
+++
@@ -75,7 +75,7 @@
    CxList SafeCookieSecurityPolicy =
        memberAccesses.FindByMemberAccesses(new string[]
        {"CookieSecurePolicy.Always", "CookieSecurePolicy.SameAsRequest"});
-
```

```
+
    /*
    The purpose of this section is to identify misconfigured Cookie policy options such as:

@@ -87,9 +87,12 @@
    It will also highlight Startup classes or configure methods where there's no configuration at all
    */
    CxList AssignedOptionField = SafeCookieSecurityPolicy.GetAssignee().FindByShortName("Secure");
-   CxList cookiSecureOption = memberAccesses.FindByMemberAccess("CookieSecureOption.Always");
-   CxList secureCookieClass = All.GetClass(cookiSecureOption);
+
+   CxList cookiSecureOption = memberAccesses.FindByMemberAccesses(new []
+       {"CookieSecureOption.Always", "CookieSecurePolicy.SameAsRequest"});
+   CxList secureCookieClass = Find_ClassDecl().GetClass(cookiSecureOption);

+
    CxList test = ASPNetCoreCookiePolicyOptions - AssignedOptionField.GetAncOfType<MethodInvokeExpr>();
    //in asp.net core we could define the security individually for each cookie
    //remove unsafe cookie when there is a CookieSecureOption.Always
```

CSharp / CSharp_Medium_Threat / Missing_Column_Encryption

Code changes

```
---
+++
@@ -62,4 +62,4 @@
 //Remove all opens that are influenced by safes
 opens -= opens.InfluencedBy(safes);
 //The result is a flow from any first param of sqlconnection to the actual openning of a connection
-result = opens.InfluencedBy(paramsConnection);
+result = opens.InfluencedBy(paramsConnection).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Medium_Threat / MVC_View_Injection

Code changes

```
---
+++
@@ -16,4 +16,4 @@
 CxList sanitizers = Find_Encode();
 sanitizers.Add(Find_Integers());

-result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Medium_Threat / Privacy_Violation

Code changes

```
---
+++
@@ -1,120 +1,95 @@
-// This query searches for variables an constants that could contain personal sensitive data,
```

```
+// This query searches for variables and constants that could contain personal sensitive data,
 // which is streamed to an output.


+// 1) Calculate inputs.
 CxList strings = Find_Strings();
-CxList integerLiteral = Find_IntegerLiterals();
-CxList nullLiteral = Find_NullLiteral();
-CxList methods = Find_Methods();
+CxList fromConfig = strings.FindByFileName("*app.config");
+
+CxList inputs = All.NewCxList(
+   Find_Inputs(),
+   Find_DB_Out(),
+   fromConfig);
+
+// If there are no inputs, we don't need to calculate anything else in this query.
+if (inputs.Count == 0)
+   return All.NewCxList();
+
+
+// 2) Calculate outputs.
+// Add exceptions (that could be thrown) to the outputs.
 CxList objectCreations = Find_ObjectCreations();
-CxList literals = All.NewCxList();
-literals.Add(strings, integerLiteral);
+CxList exceptionCreations = objectCreations.FindByName("*Exception");


-// Find names that are suspected to be personal info
-// eg. String PASSWORD, Integer SSN, and remove string literals, such as x = "password"
-CxList personal_info = Find_Personal_Info() - strings;
+// Handle the case where the super (base) constructor of the exception is used to create a new throwable exception.
+CxList exceptionCtorsWithSuper = Find_ConstructorDecl().FindByName("*Exception")
+   .FilterByDomProperty<ConstructorDecl>(c => c.BaseParameters.Count > 0);


-// 1) Exclude variables that are all uppercase - usually describes the pattern
-//    of the data, such as PASSWORDPATTERN, PASSORDTYPE...
-personal_info -= personal_info.Filter(pi => !pi.ShortName.Any(c => char.IsLower(c)));
+CxList exceptionOutputs = All.NewCxList(exceptionCreations, exceptionCtorsWithSuper);


-// 2) Exclude constants that are assigned a literal
-CxList constants = personal_info.FindByType<ConstantDecl>();
-CxList allConstRef = personal_info.FindAllReferences(constants);
-CxList allConstRefOrigin = All.NewCxList(allConstRef);
-
-// Find all assignments of string or integer literals
-CxList constAssignedL = literals.FindByFathers(allConstRef.FindByType<Declarator>());
-
-// Remove assignments of constants to string or integer literals
```

```
-allConstRef -= personal_info.FindAllReferences(constAssignedL.GetFathers());
-
-// Remove assignments of constants to null literals
-CxList declWithNull = allConstRef * nullLiteral.GetFathers().FindByType<Declarator>();
-
-// Constants are assigned null value by default if the real assignment is not
-// in the declaration line, and so the implicit assignment to null is irrelevant.
-// eg. final x; is parsed as final x=null; although x can be assigned later
-CxList allReferences = allConstRef.FindAllReferences(declWithNull);
-
-// FindByAssignmentSide(left) finds the real assignments, eg. x = ...
-CxList allReferencesAssignee = allReferences.FindByAssignmentSide(CxList.AssignmentSide.Left);
-CxList toRemove = declWithNull - allReferences.FindDefinition(allReferencesAssignee);
-allConstRef -= toRemove;
-
-// Find all assignments to constants
-CxList constAssignments = allConstRef.FindByAssignmentSide(CxList.AssignmentSide.Left).GetFathers();
-
-// Find assignments of literals to constant personal_info and remove them from results
-CxList PI_literal = literals.GetFathers() * constAssignments;
-allConstRef -= allConstRef.FindAllReferences(allConstRef.FindByFathers(PI_literal));
-
-// Remove from personal_info all references that were removed above
-personal_info -= (allConstRefOrigin - allConstRef);
-
-// Reduce personal_info to be only personal info that is influenced by input
-CxList fromConfig = strings.FindByFileName("*app.config");
-CxList inputs = Find_Inputs();
-inputs.Add(Find_DB_Out(), fromConfig);
-personal_info = personal_info.DataInfluencedBy(inputs);
-personal_info.Add(personal_info * inputs);
-
-// 3) Add exceptions (that could be thrown) to outputs.
-CxList exceptions = objectCreations.FindByName("*Exception");
-CxList exceptionsCtors = Find_ConstructorDecl().FindByName("*Exception");
-
-// Handle the case where the super (base) constructor of the exception is used to create a new throwable exception
-CxList exceptionsCtorsWithSuper = exceptionsCtors.FilterByDomProperty<ConstructorDecl>(c => c.BaseParameters.Count > 0);
-
-// Define outputs
 CxList outputs = All.NewCxList(
     Find_Outputs(),
-   exceptions,
-   exceptionsCtorsWithSuper,
     Find_Log_Outputs(),
-   Find_Cloud_Interactive_Outputs());
+   Find_Cloud_Interactive_Outputs(),
+   exceptionOutputs);
```

```
-// Define sanitize
-CxList sanitize = Find_DB_In(); // In some languages is called Find_DB, Find_DB_In, Find_DB_Input
-sanitize.Add(Find_Encrypt(), Find_Booleans());
+// If there are no outputs, we don't need to calculate anything else in this query.
+if (outputs.Count == 0)
+   return All.NewCxList();

-// Add methods that return information about the object - and not the object itself
-sanitize.Add(methods.FindByShortNames(new [] {"nameof", "typeof"}));
-sanitize.Add(All.FindByShortNames(new [] {"GetType", "GetHashCode", "Equals"}));

-// Add additional "integer" sanitizers
-CxList memberAccessesList = base.Find_MemberAccesses();
-sanitize.Add(memberAccessesList.FindByShortNames(new [] {"*Length*","*Index*","*Contains*", "*Count*"}, true));
+// 3) Calculate sanitizers.
+CxList memberAccesses = Find_MemberAccesses();
+CxList integerMembers = memberAccesses.FindByShortNames("*Length*", "*Index*", "*Contains*", "*Count*");
+CxList guidObjects = objectCreations.FindByShortName("Guid");

-// Add second argument of "Regex.Replace(input, pattern, ...)"
-sanitize.Add(All.GetParameters(methods.FindByMemberAccess("Regex.Replace"), 1));
+// Add methods that return information about the object - not the object itself.
+CxList methods = Find_Methods();
+CxList objectMethods = methods.FindByShortNames("nameof", "typeof", "GetType", "GetHashCode", "Equals");
+
+// Add second argument of "Regex.Replace(input, pattern, ...)".
+CxList regexReplaceArgs = Find_Expressions().GetParameters(methods.FindByMemberAccess("Regex.Replace"), 1);

 // Add response from web services: var response = MyWebService.Update(input)
 CxList webServiceResponses = Find_Web_Services().GetMembersOfTarget().FindByType<MethodInvokeExpr>().GetAssignee();
-sanitize.Add(webServiceResponses);

-// Add Gui objects to sanitizers
-CxList guidObjects = objectCreations.FindByShortName("Guid");
-sanitize.Add(guidObjects);
+CxList sanitize = All.NewCxList(
+   Find_DB_In(),
+   Find_Encrypt(),
+   Find_Booleans(),
+   integerMembers,
+   guidObjects,
+   objectMethods,
+   regexReplaceArgs,
+   webServiceResponses);

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

```
-// Find declarators of constants and variables so they can be removed - declarators
-// are not a part of the flow from input to output
-// eg. string x = ___ is parsed as: (Declarator) string (UnknownReference) x (AssignExpr) = (value / expression / literal)___
-// the real flow is from the UnknownReference and not the Declarator
-CxList declarator = personal_info.FindByType<Declarator>();
+// 4) Calculate personal information.
+// eg. String password / Integer ssn
+CxList personalInfo = Find_Personal_Info();

-// Remove the declaration from the references of the variables and constants
-variableRef -= declarator;
-allConstRef -= declarator;
+CxList constants = personalInfo * Find_ConstantDecl();
+CxList constantRefs = personalInfo.FindAllReferences(constants);

-// Find all constants that are assigned from an input (directly or indirectly) and are influencing an output
-CxList constInfluencedByInput = inputs.InfluencingOnAndNotSanitized(outputs, sanitize).IntersectWithNodes(allConstRef);
+CxList unknownRefs = personalInfo * Find_UnknownReference();
+CxList fieldRefs = unknownRefs.FindAllReferences(Find_FieldDecls());
+CxList propertyFieldRefs = fieldRefs.GetByAncs(Find_PropertyDecl());

-// Find all variables that are influencing an output
-CxList variableRefPath = outputs.InfluencedByAndNotSanitized(variableRef, sanitize);
+// Exclude variables that are all uppercase.
+// Usually describes the pattern of the data such as PASSWORDPATTERN, PASSWORDTYPE, ...
+CxList uppercaseInfo = personalInfo.Filter(pi => !pi.ShortName.Any(c => char.IsLower(c)));

-result = variableRefPath;
-result.Add(constInfluencedByInput);
-result = result.ReduceFlow(CxList.ReduceFlowType.ReduceSmallFlow);
+CxList exclusions = All.NewCxList(constantRefs, propertyFieldRefs, uppercaseInfo);
+CxList relevantTypes = All.NewCxList(unknownRefs, memberAccesses, methods);
+
+personalInfo *= relevantTypes - exclusions;
+
+
+// 5) Calculate the flows from inputs to outputs through personal information.
+CxList taintedPersonalInfo = inputs.InfluencingOn(personalInfo).GetLastNodesInPath();
+taintedPersonalInfo.Add(personalInfo * inputs);
+
+// Find all tainted personal information that is influencing an output.
+result = taintedPersonalInfo.InfluencingOnAndNotSanitized(outputs, sanitize)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceSmallFlow);
+
+result.Add(taintedPersonalInfo * outputs);
```

CSharp / CSharp_Medium_Threat / ReDoS_In_Validation

Code changes

```
---
+++
@@ -1,3 +1,4 @@
 CxList evilStrings = Find_Evil_Strings_Pure();

-result = evilStrings.DataInfluencingOn(All.FindByType("ValidationExpression"));
+result = evilStrings.DataInfluencingOn(All.FindByType("ValidationExpression"))
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Medium_Threat / Session_Fixation

Code changes

```
---
+++
@@ -14,6 +14,6 @@
 CxList inputs = Find_Inputs();

 CxList sessionIds = All.NewCxList(sessionIdMethods, sessionIdIndexerRefs.GetFathers());
-CxList sessionIdsInfluenced = sessionIds.InfluencedBy(inputs);
+CxList sessionIdsInfluenced = sessionIds.InfluencedBy(inputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

 result = sessionIdsInfluenced;
```

CSharp / CSharp_Medium_Threat / SSL_Verification_Bypass

Code changes

```
---
+++
@@ -76,3 +76,5 @@
     x509Certs - safeCertFlows,
     certCheckResults,
     delegates.InfluencingOnAndNotSanitized(delegateTargets, sanitizers));
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Medium_Threat / SSRF

Code changes

```
---
+++
@@ -158,9 +158,10 @@
 /*General sanitizers
 e.g. hashing, encrypting, integers and casting to int/bool/float
 */
-CxList generalSanitizers = Find_Hashing_Functions();
-generalSanitizers.Add(Find_Encrypt());
-generalSanitizers.Add(Find_Integers());
+CxList generalSanitizers = All.NewCxList(
+   Find_Hashing_Functions(),
+   Find_Encrypt(),
+   Find_Integers());
```

```
 //Casts to int/bool/float
 List<string> strType = new List<string> {"int", "Int16", "Int32", "Int64", "UInt64",
@@ -195,5 +196,7 @@
 //Add to results
 result.Add(socketPaths);

+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
+
 // When the flow goes through an "Remote input" (a request to a remote server) it is not vulnerable
 result -= result.IntersectWithNodes(Find_Remote_Requests());
```

CSharp / CSharp_Medium_Threat / Stored_Command_Injection

Code changes

```
---
+++
@@ -7,4 +7,4 @@
 CxList exec = Find_Command_Execution();
 CxList sanitize = Find_Sanitize();

-result = exec.InfluencedByAndNotSanitized(inputs, sanitize);
+result = exec.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Medium_Threat / Use_of_Cryptographically_Weak_PRNG

Code changes

```
---
+++
@@ -47,4 +47,4 @@
     Find_Decrypt(),
     unknownReferences.GetParameters(refCompHashAndHashCore));

-result = inputs.InfluencingOn(outputs);
+result = inputs.InfluencingOn(outputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Medium_Threat / Use_of_Hard_coded_Cryptographic_Key

Code changes

```
---
+++
@@ -148,3 +148,4 @@
         }
     }
 }
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_WebConfig / TraceEnabled

Code changes

```
---
+++
```

```
@@ -1,17 +1,24 @@
 CxList webConfig = Find_Web_Config();
+
+// Find all string literals representing "true" and "false" within the webConfig
 CxList value_false = webConfig.FindByName("false").FindByType<StringLiteral>();
 CxList value_true = webConfig.FindByName("true").FindByType<StringLiteral>();

-CxList enabledTrue = value_true.DataInfluencingOn(webConfig.FindByName("CONFIGURATION.SYSTEM.WEB.TRACE.ENABLED"));
+CxList traceConfig = webConfig.FindByName("CONFIGURATION.SYSTEM.WEB.TRACE.ENABLED");
+
+// Find configurations influencing traceEnabled and localOnly settings
+CxList traceEnabledTrue = value_true.DataInfluencingOn(traceConfig);
+CxList traceEnabledFalse = value_false.DataInfluencingOn(traceConfig);
 CxList localOnlyFalse = value_false.DataInfluencingOn(webConfig.FindByName("CONFIGURATION.SYSTEM.WEB.TRACE.LOCALONLY"));

-CxList enabTrueAndLocalOnlyFalse = All.NewCxList(enabledTrue, localOnlyFalse);
-if (enabTrueAndLocalOnlyFalse.Count > 1)
+// Filter values influencing traceEnabled and localOnly settings
+CxList traceTrueAndLocalOnlyFalse = All.NewCxList(traceEnabledTrue, localOnlyFalse);
+CxList traceFalseAndLocalOnlyFalse = All.NewCxList(traceEnabledFalse, localOnlyFalse);
+
+if (traceTrueAndLocalOnlyFalse.Count >= 1)
 {
-   result = value_true * enabledTrue;
+   result = value_true * traceEnabledTrue;
+   if(traceFalseAndLocalOnlyFalse.Count < 2) {
+       result.Add(value_false * localOnlyFalse);
+   }
 }
-if (enabTrueAndLocalOnlyFalse.Count == 1)
-{
-   result = value_true * enabledTrue;
-   result.Add(value_false * localOnlyFalse);
-}
```

CSharp / CSharp_Windows_Phone / Client_Side_Injection

Code changes

```
---
+++
@@ -15,7 +15,7 @@
 CxList sanitize = Find_Sanitize();

 // Find SQL injection
-result = db.InfluencedByAndNotSanitized(inputs, sanitize);
+result = db.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

 // Find XSS is not implemented at this stage.
 //CxList xssSanitize = Find_XSS_Sanitize();
```

CSharp / CSharp_Windows_Phone / Hard_Coded_Cryptography_Key

Code changes

```
---
+++
@@ -41,4 +41,4 @@
 CxList notHardcoded = All.NewCxList(sanitizedHardCoded, initValues, edgeCase);
 hardCoded -= notHardcoded;

-result = hardCoded.DataInfluencingOn(keys);
+result = hardCoded.DataInfluencingOn(keys).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Windows_Phone / Insecure_Data_Storage

Code changes

```
---
+++
@@ -14,4 +14,4 @@
 CxList writes = Find_Write();
 CxList encrypt = Find_Encrypt();

-result = writes.InfluencedByAndNotSanitized(passwords, encrypt);
+result = writes.InfluencedByAndNotSanitized(passwords, encrypt).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

CSharp / CSharp_Windows_Phone / Poor_Authorization_and_Authentication

Code changes

```
---
+++
@@ -17,4 +17,4 @@
 CxList write = Find_Write();
 CxList outInfluencedByHttp = write * write.DataInfluencedBy(http);

-result = deviceId.DataInfluencingOn(outInfluencedByHttp);
+result = deviceId.DataInfluencingOn(outInfluencedByHttp).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);;
```

CSharp / CSharp_Windows_Phone / Side_Channel_Data_Leakage

Code changes

```
---
+++
@@ -6,4 +6,4 @@
 CxList personal = All.NewCxList(Find_All_Passwords(), Find_Personal_Info());
 CxList sanitize = Find_Encrypt();

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

Dart / Dart_Mobile_High_Risk / Resource_Updated_By_URL_Data

Code changes

```
---
+++
```

```
@@ -7,4 +7,4 @@
 // Any form of decryption is considered a sanitizer
 CxList sanitizers = All.FindByShortNames("*decrypt*", "*unencrypt*");

-result = sinks.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = sinks.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Dart / Dart_Mobile_Low_Visibility / Missing_Certificate_Pinning

Code changes

```
---
+++
@@ -29,3 +29,5 @@
 CxList httpSinks = Find_Remote_Sinks();
 result = httpSinks.InfluencedBy(dirtySecurityContext);
 result.Add(httpSinks.NotInfluencedBy(securityContext));
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Dart / Dart_Mobile_Low_Visibility / Private_Storage_WebView_JavaScript_Injection

Code changes

```
---
+++
@@ -8,4 +8,4 @@
 CxList sanitizedSinks = sinks.InfluencedBy(sanitizedMimeTypes).GetLastNodesInPath();
 sinks -= sanitizedSinks;

-result = sinks.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = sinks.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Dart / Dart_Mobile_Low_Visibility / Self_SQL_Injection

Code changes

```
---
+++
@@ -7,4 +7,4 @@
 //Outputs
 CxList outputs = Find_DB_In();

-result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Dart / Dart_Mobile_Low_Visibility / Self_WebView_JavaScript_Injection

Code changes

```
---
+++
@@ -8,4 +8,4 @@
 CxList sanitizedSinks = sinks.InfluencedBy(sanitizedMimeTypes).GetLastNodesInPath();
 sinks -= sanitizedSinks;
```

```
-result = sinks.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = sinks.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Dart / Dart_Mobile_Low_Visibility / Unencrypted_Sensitive_Information_in_Temporary_File

Code changes

```
---
+++
@@ -22,4 +22,4 @@

 CxList outputs = writeMethods.InfluencedByAndNotSanitized(sensitiveInfo, encryHash).GetLastNodesInPath();

-result = outputs.InfluencedBy(inputs);
+result = outputs.InfluencedBy(inputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Dart / Dart_Mobile_Low_Visibility / User_Information_in_Publicly_Accessible_Storage

Code changes

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

     CxList sinks = Find_Firebase_Cloud_Sinks();

-    result = sinks.InfluencedByAndNotSanitized(inputs, sanitizers);
+    result = sinks.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

Dart / Dart_Mobile_Medium_Threat / Absolute_Path_Traversal

Code changes

```
---
+++
@@ -20,4 +20,4 @@
 CxList sanitizedTargets = unknowns.FindAllReferences(sanitizedInputs);
 CxList allSanitizedInputs = inputs.GetMembersWithTargets(sanitizedTargets);
 CxList unsanitizedInputs = inputs - allSanitizedInputs;
-result.Add(sinks.InfluencedBy(unsanitizedInputs));
+result = sinks.InfluencedBy(unsanitizedInputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Dart / Dart_Mobile_Medium_Threat / Insecure_Asymmetric_Cryptographic_Algorithm_Parameters

Code changes

```
---
+++
@@ -6,4 +6,4 @@

 CxList parseMethod = methods.FindByMemberAccess("RSAKeyParser.parse");

-result = parseMethod.InfluencedBy(inputStrings);
+result = parseMethod.InfluencedBy(inputStrings).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Dart / Dart_Mobile_Medium_Threat / Insecure_WebSocket_Connection

Code changes

```
---

+++

@@ -6,4 +6,4 @@


 CxList webSocketConnect = webSockets.GetMembersOfTarget().FindByShortName("connect");


-result = webSocketConnect.InfluencedBy(insecureWebSocketProtocol);

+result = webSocketConnect.InfluencedBy(insecureWebSocketProtocol).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Dart / Dart_Mobile_Medium_Threat / Poor_Authorization_and_Authentication

Code changes

```
---

+++

@@ -2,4 +2,4 @@


 CxList outputs = Find_Remote_Sinks();


-result = inputs.InfluencingOn(outputs);

+result = inputs.InfluencingOn(outputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Dart / Dart_Mobile_Medium_Threat / Public_Storage_SQL_Injection

Code changes

```
---

+++

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

 //Outputs

 CxList outputs = Find_DB_In();


-result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers);

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

Dart / Dart_Mobile_Medium_Threat / Public_Storage_WebView_JavaScript_Injection

Code changes

```
---

+++

@@ -8,4 +8,4 @@

 CxList sanitizedSinks = sinks.InfluencedBy(sanitizedMimeTypes).GetLastNodesInPath();

 sinks -= sanitizedSinks;


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

+result = sinks.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Dart / Dart_Mobile_Medium_Threat / Relative_Path_Traversal

Code changes

```
---

+++
```

```
@@ -2,12 +2,13 @@
 CxList unknowns = Find_UnknownReference();
 CxList stmts = Find_LambdaExpr();
 stmts.Add(Find_ForEachStmt());
+CxList binExprs = Find_BinaryExpr();

 // Inputs
 CxList inputs = Find_UI_Inputs();

 // Concats
-CxList concats = Find_BinaryExpr().GetByBinaryOperator(BinaryOperator.Add);
+CxList concats = binExprs.GetByBinaryOperator(BinaryOperator.Add);

 // Sinks
 CxList allSinks = Find_Path_Traversal_Sinks();
@@ -15,7 +16,7 @@

 // 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));
+sinksWithConcatenatedStrings.Add(sinks.Where(sink => binExprs.GetParameters(sink, 0).Count != 0));
 sinks -= sinksWithConcatenatedStrings;

 // Sanitizers
@@ -29,3 +30,5 @@
 CxList unsanitizedInputs = inputs - allSanitizedInputs;
 result = sinksWithConcatenatedStrings.InfluencedByAndNotSanitized(unsanitizedInputs, concatenatedStringsSanitizers);
 result.Add(sinks.InfluencedBy(unsanitizedInputs));
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Dart / Dart_Mobile_Medium_Threat / SQL_Injection_from_URL_Scheme_or_Intent

Code changes

```
---
+++
@@ -8,4 +8,4 @@

 CxList sinks = Find_DB_In();

-result = sinks.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = sinks.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Dart / Dart_Mobile_Medium_Threat / Use_of_Hardcoded_Cryptographic_IV

Code changes

```
---
+++
@@ -3,4 +3,4 @@
 CxList outputs = Find_Methods().FindByMemberAccesses(new string[]{"IV", "IV.fromBase16", "IV.fromBase64",
```

```
    "IV.fromLength", "IV.fromUtf8"});


-result = outputs.InfluencedBy(hardcodedValues);
+result = outputs.InfluencedBy(hardcodedValues).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Dart / Dart_Mobile_Medium_Threat / Use_of_Hardcoded_Salt

Code changes

```
---
+++
@@ -4,5 +4,5 @@
 // Only the second parameter (salt parameter) of the kdf functions is relevant.
 CxList outputs = All.GetParameters(kdf, 1);

-result = outputs.InfluencedBy(hardcodedValues);
+result = outputs.InfluencedBy(hardcodedValues).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 result.Add(outputs * hardcodedValues);
```

Dart / Dart_Mobile_Medium_Threat / WebView_JavaScript_Injection_from_URL_Scheme

Code changes

```
---
+++
@@ -1,5 +1,3 @@
-CxList unknRefs = Find_UnknownReference();
-
 CxList inputs = Find_Native_URL_Schemes_Inputs();
 inputs.Add(Find_Non_Native_URL_Schemes_Inputs());


@@ -10,4 +8,4 @@
 CxList sanitizedSinks = sinks.InfluencedBy(sanitizedMimeTypes).GetLastNodesInPath();
 sinks -= sanitizedSinks;

-result = sinks.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = sinks.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Go / Go_AWS_Lambda / AWS_Credentials_Leak

Code changes

```
---
+++
@@ -55,3 +55,5 @@
 CxList osGetEnvMethods = methodDecls.FindByName("os.Getenv");
 CxList osEnvVars = osGetEnvMethods.InfluencedBy(envVars).GetLastNodesInPath();
 result.Add(outputs.InfluencedBy(osEnvVars));
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Go / Go_AWS_Lambda / Hardcoded_AWS_Credentials

Code changes

```
---
+++
@@ -6,7 +6,7 @@
 CxList vulnerableAwsApi = awsApi.FindByMemberAccesses(new string[] {"aws.*", "credentials.*"}, false);


 //Find vulnerable methods and object creation
-CxList vulnerableMethodsObj = vulnerableAwsApi.FindByShortNames(new List<string>
+CxList vulnerableMethodsObj = vulnerableAwsApi.FindByShortNames(new []
    {"NewStaticCredentialsProvider",
        "StaticCredentialsProvider",
        "Credentials"});
@@ -28,4 +28,4 @@
 //Get relevant strings
 CxList relevantStrings = strings - unnecessaryParamsValue;

-result = vulnerableMethodsObj.InfluencedBy(relevantStrings);
+result = vulnerableMethodsObj.InfluencedBy(relevantStrings).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Go / Go_AWS_Lambda / Unrestricted_Write_S3

Code changes

```
---
+++
@@ -16,5 +16,5 @@
 outputs -= outputs.GetByAncs(s3Types.InfluencedBy(headObjectParams).GetLastNodesInPath());


-result = inputs.InfluencingOn(outputs);
+result = inputs.InfluencingOn(outputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 result.Add(inputs * outputs);
```

Go / Go_AWS_Lambda / Use_of_Hardcoded_Cryptographic_Key_On_Server

Code changes

```
---
+++
@@ -25,7 +25,7 @@
     .CxSelectDomProperty<Param>(_ => _.Value);


 // Flows from string literals to parameters
-result.Add(keyParams.InfluencedBy(strings));
+result.Add(keyParams.InfluencedBy(strings).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow));


 // Hardcoded strings in the parameter
 result.Add(keyParams * strings);
```

Go / Go_High_Risk / Connection_String_Injection

Code changes

```
---
+++
```

```
@@ -15,4 +15,4 @@
    Find_WhiteListSanitizers(),
    hashAndEncrypt);

-result = inputs.InfluencingOnAndNotSanitized(outputs, sanitizers);
+result = inputs.InfluencingOnAndNotSanitized(outputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Go / Go_High_Risk / Unsafe_Reflection

Code changes

```
---
+++
@@ -28,4 +28,4 @@

 CxList validOutputs = All.FindByParameters(validResults.GetLastNodesInPath());

-result = validOutputs.InfluencedBy(inputs);
+result = validOutputs.InfluencedBy(inputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Go / Go_Low_Visibility / Empty_Password_In_Connection_String

Code changes

```
---
+++
@@ -13,4 +13,4 @@
 //sql.Open method second parameter is the connection string
 CxList connectionString = Find_Database_Open_Params();

-result = emptyPasswordParam.InfluencingOn(connectionString);
+result = emptyPasswordParam.InfluencingOn(connectionString).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Go / Go_Low_Visibility / Open_Redirect

Code changes

```
---
+++
@@ -33,4 +33,4 @@
 CxList sanitizedRefs = unkRefs.FindAllReferences(relevantArgumentsReferencesSanitized);
 sanitizers.Add(sanitizedRefs);

-result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Go / Go_Medium_Threat / Divide_By_Zero

Code changes

```
---
+++
@@ -150,3 +150,5 @@
 result.Add(zero.InfluencingOnAndNotSanitized(divWithoutIf, sanitize),
    inputs.DataInfluencingOn(divWithoutIf).SanitizeCxList(sanitize),
    inputs * divBin.CxSelectDomProperty<BinaryExpr>(_ => _.Right)); //If the inputs are on the right side of the div expr directly, add them to the results
```

```
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Go / Go_Medium_Threat / Email_Content_Forgery

Code changes

```
---
+++
@@ -3,25 +3,25 @@
 CxList inputs = Find_Interactive_Inputs();
 inputs.Add(Find_CGI_Inputs());

-CxList sanitizers = methods.FindByMemberAccess("html.EscapeString");
-string[] hashMethods = new string[]{"argon2.HashEncoded","argon2.HashRaw","argon2.Hash"};
-sanitizers.Add(methods.FindByMemberAccesses(hashMethods, false));
-sanitizers.Add(methods.FindByMemberAccess("base64.NewDecoder"));
-sanitizers.Add(Find_General_Sanitize());
+CxList sanitizers = methods.FindByMemberAccesses(new [] {"html.EscapeString", "base64.NewDecoder"});
+sanitizers.Add(
+   //Hash methods
+   methods.FindByMemberAccesses(new []{"argon2.HashEncoded","argon2.HashRaw","argon2.Hash"}, false),
+   Find_General_Sanitize());

 CxList sendMailMethod = methods.FindByMemberAccess("smtp.SendMail");
 CxList sendMailMethodFifthParam = All.GetParameters(sendMailMethod, 4);
-CxList outputs = sendMailMethodFifthParam;
+CxList outputs = All.NewCxList(sendMailMethodFifthParam);
 outputs.Add(unkRefs.GetByAncs(sendMailMethodFifthParam));

 CxList dial = methods.FindByMemberAccess("smtp.Dial");
 CxList data = methods.FindByShortName("Data");
-CxList dataBody = data.DataInfluencedBy(dial).GetStartAndEndNodes(CxList.GetStartEndNodesType.EndNodesOnly);
+CxList dataBody = data.DataInfluencedBy(dial).GetLastNodesInPath();
 outputs.Add(dataBody);

 CxList dataBodyRefs = unkRefs.FindAllReferences(dataBody.GetAssignee(0));
-CxList writeTo = methods.FindByShortName("WriteTo").InfluencedBy(inputs).GetStartAndEndNodes(CxList.GetStartEndNodesType.EndNodesOnly);
+CxList writeTo = methods.FindByShortName("WriteTo").InfluencedBy(inputs).GetLastNodesInPath();
 CxList relevantWriteTo = dataBodyRefs.GetByAncs(writeTo);
-outputs.Add(relevantWriteTo.GetAncOfType(typeof(MethodInvokeExpr)));
+outputs.Add(relevantWriteTo.GetAncOfType<MethodInvokeExpr>());

-result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Go / Go_Medium_Threat / Hardcoded_Password_in_Connection_String

Code changes

```
---
+++
```

```
@@ -9,4 +9,4 @@
 CxList methodSndParam = Find_Database_Open_Params();


 result = strLiterals * methodSndParam;
-result.Add(regEx.DataInfluencingOn(methodSndParam));
+result.Add(regEx.DataInfluencingOn(methodSndParam).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow));
```

Go / Go_Medium_Threat / Insecure_Value_of_the_SameSite_Cookie_Attribute_in_Code

Code changes

```
---
+++
@@ -117,3 +117,5 @@
     ginSetCookie.InfluencedBy(ginSetSameSite.InfluencedBy(unsafeValues).GetFirstNodesInPath()),
     ginSetCookie.NotInfluencedBy(ginSetSameSite),
     relevantHeaderMethods.InfluencedBy(unsafeStrings));
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Android / Client_Side_Injection

Code changes

```
---
+++
@@ -71,4 +71,4 @@
 CxList sanitizers = Find_Sanitize();
 sanitizers.Add(Find_GetInitParameter_Method());


-result = db.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = db.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Android / Implicit_Intent_With_Read_Write_Permissions

Code changes

```
---
+++
@@ -23,4 +23,4 @@
         "startService",
         "startActivities"});


-result = startActivity.InfluencedByAndNotSanitized(addedReadWrite, sanitize);
+result = startActivity.InfluencedByAndNotSanitized(addedReadWrite, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Android / Information_Leak_Through_Response_Caching

Code changes

```
---
+++
@@ -41,6 +41,7 @@
                 IgnorCacheSanitizer.Add(IgnorCacheHeader.GetTargetOfMembers());
             }
         }
```

```
-       result = HttpResponse.InfluencedByAndNotSanitized(HttpRequest, IgnorCacheSanitizer);
+       result = HttpResponse.InfluencedByAndNotSanitized(HttpRequest, IgnorCacheSanitizer)
+           .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
    }
 }
```

Java / Java_Android / Insecure_Data_Storage

Code changes

```
---
+++
@@ -19,4 +19,4 @@

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

-result = allWrites.DataInfluencedBy(sd);
+result = allWrites.DataInfluencedBy(sd).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Android / Insecure_WebView_Usage

Code changes

```
---
+++
@@ -66,7 +66,7 @@
    CxList sanitizer = All.GetByAncs(markerStmtCol); // the closure of these if-blocks is sanitized
    CxList loadUrl = webViews.GetMembersOfTarget().FindByShortName("loadUrl");
    CxList inputs = Find_Interactive_Inputs();
-   result = loadUrl.InfluencedByAndNotSanitized(inputs, sanitizer);
+   result = loadUrl.InfluencedByAndNotSanitized(inputs, sanitizer).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }

 ////WebSettings.setPluginState
```

Java / Java_Android / Insufficient_Application_Layer_Protect

Code changes

```
---
+++
@@ -16,5 +16,5 @@
    CxList pureHTTP = Find_Pure_http();

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

Java / Java_Android / Insufficient_Sensitive_Application_Layer

Code changes

```
---
+++
@@ -15,6 +15,7 @@
```

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


@@ -45,7 +46,7 @@
    //support OKHttpClient
    CxList okHttpClient = methods.FindByMemberAccess("OkHttpClient.newCall");

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

Java / Java_Android / Poor_Authorization_and_Authentication

Code changes

```
---
+++
@@ -9,4 +9,4 @@
 CxList findWrite = Find_Write();
 CxList outInfluencedByHttp = findWrite * findWrite.DataInfluencedBy(http);
 CxList deviceIDInfo = Find_Methods().FindByMemberAccess("TelephonyManager.getDeviceId");
-result = deviceIDInfo.DataInfluencingOn(outInfluencedByHttp);
+result = deviceIDInfo.DataInfluencingOn(outInfluencedByHttp).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Android / Use_Of_Implicit_Intent_For_Sensitive_Communication

Code changes

```
---
+++
@@ -34,4 +34,4 @@
 // Find the call to the activity using the implicit intent
 CxList sentIntent = activity.InfluencedBy(implicitIntent).GetLastNodesInPath();

-result = sentIntent.InfluencedBy(sensitiveInfo);
+result = sentIntent.InfluencedBy(sensitiveInfo).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_AWS_Lambda / AWS_Credentials_Leak

Code changes

```
---
+++
@@ -27,4 +27,4 @@
    "AWS_SESSION_TOKEN");

 CxList credInfoAndEnvVars = All.NewCxList(credentialsInfo, envVars);
-result.Add(outputs.InfluencedBy(credInfoAndEnvVars));
```

```
+result.Add(outputs.InfluencedBy(credInfoAndEnvVars).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow));
```

Java / Java_AWS_Lambda / DynamoDB_NoSQL_Injection

Code changes

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

 CxList outputs = DynamoDB_Find_DB_In();

-result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_AWS_Lambda / Hardcoded_AWS_Credentials

Code changes

```
---
+++
@@ -9,7 +9,7 @@
 CxList keyParams = All.GetParameters(awsConstructor);

 // Flows from String literals to parameters
-result = keyParams.InfluencedBy(strings);
+result = keyParams.InfluencedBy(strings).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

 // Hardcoded strings in the parameter
 result.Add(keyParams * strings);
```

Java / Java_AWS_Lambda / Permission_Manipulation_in_S3

Code changes

```
---
+++
@@ -23,4 +23,4 @@
 CxList relevantParams = inputs.InfluencingOn(unknownRefsParamsOut);
 relevantParams.Add(relevantMethodsInfluencedByInputs.GetAncOfType<LambdaExpr>().GetAssignee());

-result = s3APIOut.InfluencedBy(relevantParams);
+result = s3APIOut.InfluencedBy(relevantParams).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_AWS_Lambda / Unrestricted_Delete_S3

Code changes

```
---
+++
@@ -12,4 +12,4 @@
 //Get Sanitizers
 CxList sanitizers = inputs.InfluencingOn(relevantRefsConditions);

-result = s3API.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = s3API.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_AWS_Lambda / Unrestricted_Write_S3

Code changes

```
---
+++
@@ -14,4 +14,4 @@
 //Get Sanitizers
 CxList sanitizers = inputs.InfluencingOn(relevantRefsConditions);

-result = s3Outs.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = s3Outs.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_AWS_Lambda / User_Based_SDK_Configurations

Code changes

```
---
+++
@@ -23,7 +23,7 @@
     "region"});
 CxList relMethods = relMethodsNames.GetMembersWithTargets(s3Client, -1);

-result.Add(relMethods.InfluencedByAndNotSanitized(inputs, sanitizers));
+result.Add(relMethods.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow));

 // modifyRequest Method
 CxList modifyRequest = methods.FindByShortName("modifyRequest");
```

Java / Java_AWS_Lambda / Use_of_Hardcoded_Cryptographic_Key_On_Server

Code changes

```
---
+++
@@ -26,7 +26,7 @@
 sanitizers.Add(Find_FileSystem_ReadMethods());

 // Flows from string literals to parameters
-result.Add(parameters.InfluencedByAndNotSanitized(strings,sanitizers));
+result.Add(parameters.InfluencedByAndNotSanitized(strings,sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow));

 // Hardcoded strings in the parameter
 result.Add(parameters * strings);
```

Java / Java_Best_Coding_Practice / Incorrect_Conversion_between_Numeric_Types

Code changes

```
---
+++
@@ -72,7 +72,7 @@
 sqrtParam -= sqrtParam.FindAllReferences(paramInCondition);
 CxList input = Find_Interactive_Inputs();

-result.Add(sqrtParam.DataInfluencedBy(input));
```

```
+result.Add(sqrtParam.DataInfluencedBy(input).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow));


 /// Remove dead code
 result -= deadCodeContents;
```

Java / Java_Best_Coding_Practice / Reliance_On_Untrusted_Inputs_In_Security_Decision

Code changes

```
---
+++
@@ -27,7 +27,7 @@
 conditions = conditions * relevant_objects;


 // first part. path from input to conditions with sanitizer
-CxList part1 = inputs.InfluencingOnAndNotSanitized(conditions, sanitizer);
+CxList part1 = inputs.InfluencingOnAndNotSanitized(conditions, sanitizer).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);


 //second part. path from conditions (ifs) to get all variables inside to sinks
 CxList sinksInIfs = sink.GetByAncs(ifAllCond);
```

Java / Java_GWT / GWT_DOM_XSS

Code changes

```
---
+++
@@ -12,5 +12,5 @@
         All.FindByName("*encode*", false),
         Find_Methods().FindByShortName("toSafeHtml"));


-   result = inputs.InfluencingOnAndNotSanitized(outputs, sanitize);
+   result = inputs.InfluencingOnAndNotSanitized(outputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

Java / Java_GWT / GWT_Reflected_XSS

Code changes

```
---
+++
@@ -11,5 +11,5 @@
     // only check Reflected XSS outputs
     outputs = outputs.GetByMethod(Find_MethodDecls().FindByShortName("onSuccess"));


-   result = outputs.DataInfluencedBy(inputs);
+   result = outputs.DataInfluencedBy(inputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

Java / Java_High_Risk / Expression_Language_Injection_EL

Code changes

```
---
+++
@@ -1,6 +1,6 @@
```

```
 CxList methods = Find_Methods();
+CxList inputs = Find_Interactive_Inputs();

-CxList inputs = Find_Interactive_Inputs();
 CxList isValidOverrides = Find_CustomAttribute().FindByShortName("Override").GetFathers().
    FindByType<MethodDecl>().FindByShortName("isValid");
 inputs.Add(Find_ParamDecl().GetParameters(isValidOverrides, 0));
@@ -8,7 +8,8 @@
 CxList evaluations = methods.FindByMemberAccesses(new string[] {"ELProcessor.eval",
    "ConstraintValidatorContext.buildConstraintViolationWithTemplate"});

-CxList containsParameters = All.GetParameters(Find_Conditions().FindByMemberAccess("string.contains"));
+CxList containsInvokes = Find_Conditions().FindByMemberAccesses(new string[] {"string.contains","list.contains"});
+CxList containsParameters = All.GetParameters(containsInvokes);
 CxList sanitizers = Find_UnknownReference().FindAllReferences(containsParameters);

-result.Add(inputs.InfluencingOnAndNotSanitized(evaluations, sanitizers));
+result.Add(inputs.InfluencingOnAndNotSanitized(evaluations, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow));
```

Java / Java_High_Risk / Reflected_XSS_All_Clients

Code changes

```
---
+++
@@ -72,4 +72,4 @@
 // When the flow goes through a "Remote input" (a request to a remote server) it is not vulnerable
 CxList remoteInputs = All.NewCxList(Find_Remote_ReadMethods(), Find_Remote_Connections());
-result -= result.IntersectWithNodes(remoteInputs);
+result = result.SanitizeCxList(remoteInputs);
```

Java / Java_High_Risk / Second_Order_SQL_Injection

Code changes

```
---
+++
@@ -1,6 +1,7 @@
+// 1) Calculate inputs.
 CxList dbOut = Find_DB_Out();
-
-CxList read = All.NewCxList(
+CxList inputs = All.NewCxList(
+   dbOut,
    Find_Read_NonDB(),
    Find_FileSystem_Read(),
    Find_Files_Open(),
@@ -8,53 +9,57 @@
    Find_Vulnerable_Io_File_Methods(),
    Find_Cloud_Storage_In());
```

```
+// If there are no inputs, we don't need to calculate anything else in this query.
+if (inputs.Count == 0)
+   return All.NewCxList();
+
+
+// 2) Calculate outputs.
 CxList dbIn = Find_SQL_DB_In();
-
-CxList sanitized = Find_SQL_Sanitize();
-
-CxList stringManip = Find_Stored_String_Manipulation();
-
-CxList dbParams = All.GetParameters(dbIn);
-
+CxList expressions = Find_Expressions();
+CxList dbParams = expressions.GetParameters(dbIn);
 CxList dbWithParams = dbIn.FindByParameters(dbParams);
 CxList dbWithNoParams = dbIn - dbWithParams;


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


-CxList dbOutRead = All.NewCxList(dbOut, read);
+// If there are no outputs, we don't need to calculate anything else in this query.
+if (outputs.Count == 0)
+   return All.NewCxList();


-CxList unknRefs = Find_UnknownReference();
-CxList xml = All.FindByFileName("*.xml");
-CxList cardinalSignValue = xml.FindByType<UnknownReference>().GetParameters(xml.FindByMemberAccess("*.setObject"), 1);


-CxList methods = Find_Methods();
+// 3) Calculate sanitizers.
+CxList createQueryMembers = Find_Methods().FindByMemberAccess("EntityManager.createQuery").GetTargetOfMembers();
+CxList sessionHandleRefs = outputs.FindByType("SessionHandle");
+CxList dbOutMemberRefs = dbOut.GetRightmostMember().GetAssignee();
+CxList outputRefs = outputs.GetLeftmostTarget().FindAllReferences(dbOutMemberRefs);


-sanitized.Add(
-   //remove SessionHandle from possible sinks
-   endDB.FindByType("SessionHandle"),
-   //remove from endDB instances of the same dbOut or reads
-   endDB.GetLeftmostTarget().FindAllReferences(dbOut.GetRightmostMember().GetAssignee()),
-   methods.FindByMemberAccess("EntityManager.createQuery").GetTargetOfMembers(),
-   unknRefs.FindAllReferences(All.FindByParameters(cardinalSignValue).GetTargetOfMembers()));
+CxList sanitizers = All.NewCxList(
+   createQueryMembers,
+   Find_SQL_Sanitize(),
```

```
+   sessionHandleRefs,
+   outputRefs);

-CxList allResults = dbOutRead.InfluencingOnAndNotSanitized(endDB, sanitized).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

-//Remove the flows from the final result that don't have any string manipulation
-result = allResults - (allResults.SanitizeCxList(stringManip).GetFirstNodesInPath());
+// 4) Calculate the flows from inputs to outputs that are not being sanitized.
+result = inputs.InfluencingOnAndNotSanitized(outputs, sanitizers)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

-CxList xmlFileResults = result.FindByFileName("*.xml");
+if (result.Count == 0)
+   return All.NewCxList();

-foreach(CxList res in xmlFileResults){
-   if(methods.GetMethod(res.GetFirstNodesInPath()) == methods.GetMethod(res.GetLastNodesInPath())){
-       result -= res;
-   }
-}

-//Remove results that intersect with prepared statements that weren't influenced by inputs
-CxList prepStmts = All.FindByTypes(new string[]{"PreparedStatement*", "*.PreparedStatement" });
+// Remove the flows from the final result that don't have any string manipulation.
+result -= result.SanitizeCxList(Find_Stored_String_Manipulation());
+if (result.Count == 0)
+   return All.NewCxList();
+
+// Remove results that intersect with prepared statements that were not influenced by the inputs.
+CxList prepStmts = expressions.FindByTypes(new string[]{"PreparedStatement*", "*.PreparedStatement" });
 CxList prepStmtMethods = prepStmts.GetAssigner();
-CxList prepStmtParamsInfInputs = All.GetParameters(prepStmtMethods, 0).InfluencedBy(dbOutRead).GetLastNodesInPath();
-CxList prepStmtMethodsInfInp = prepStmtMethods.FindByParameters(prepStmtParamsInfInputs);
+CxList prepStmtArgs = expressions.GetParameters(prepStmtMethods, 0);
+CxList influencedPrepStmtArgs = inputs.InfluencingOn(prepStmtArgs).GetLastNodesInPath();
+CxList influencedPrepStmtMethods = prepStmtMethods.FindByParameters(influencedPrepStmtArgs);

-CxList prepStmtMethNotInf = prepStmtMethods - prepStmtMethodsInfInp;
-result -= result.IntersectWithNodes(prepStmtMethNotInf);
+CxList notInfluencedPrepStmts = prepStmtMethods - influencedPrepStmtMethods;
+result -= result.IntersectWithNodes(notInfluencedPrepStmts);
```

Java / Java_High_Risk / Stored_XSS

Code changes

```
---
+++
@@ -1,17 +1,6 @@
-CxList readNonDB = Find_Read_NonDB();
```

```
-
-//System.getProperties should not be considered as input
-CxList listSystemGetPropertiesInInputs = readNonDB.FindByMemberAccesses(
-   new string [] {"System.getProperty","System.getProperties"});
-readNonDB -= listSystemGetPropertiesInInputs;
-
-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 inputs = All.NewCxList(
    Find_DB_Out(),
-   read,
+   Find_Read_NonDB(),
    Find_Vulnerable_Nio_Files_Methods(),
    Find_Vulnerable_Io_File_Methods(),
    Find_Cloud_Storage_In());
```

Java / Java_High_Risk / Unsafe_JNDI_Lookup

Code changes

```
---
+++
@@ -3,4 +3,5 @@
 CxList vulnerableMethods = Find_Methods().FindByMemberAccesses(new string[]
    {"DirContext.lookup", "JndiManager.lookup"});

-result = inputs.InfluencingOnAndNotSanitized(All.GetParameters(vulnerableMethods), Find_Redirect_Sanitizers());
+result = inputs.InfluencingOnAndNotSanitized(All.GetParameters(vulnerableMethods), Find_Redirect_Sanitizers())
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_High_Risk / Unsafe_Reflection

Code changes

```
---
+++
@@ -25,4 +25,4 @@
 CxList binaryExprs = Find_Strings().GetFathers().FindByType<BinaryExpr>();
 sanitize.Add(binaryExprs);

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

Java / Java_Low_Visibility / Authorization_Bypass_Through_User_Controlled_SQL_PrimaryKey

Code changes

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

```
 /// DB influenced by potentially problematic input
-result = Find_DB_In().DataInfluencedBy(input).DataInfluencedBy(id);
+result = Find_DB_In().DataInfluencedBy(input).DataInfluencedBy(id).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Low_Visibility / Collapse_of_Data_into_Unsafe_Value

Code changes

```
---
+++
@@ -18,3 +18,4 @@
 CxList relevantReplaceInLoop = relevantReplace.GetByAncs(loops);


 result = relevantReplace - relevantReplaceInLoop;
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Low_Visibility / Cookie_Overly_Broad_Path

Code changes

```
---
+++
@@ -35,4 +35,4 @@
 // setPath will now include only paths that are no concatenated
 setPath -= concatSetPath;


-result = setPath.InfluencedBy(rootPath);
+result = setPath.InfluencedBy(rootPath).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Low_Visibility / Creation_of_Temp_File_in_Dir_with_Incorrect_Permissions

Code changes

```
---
+++
@@ -10,3 +10,4 @@
 CxList createInfluenced = newFileObject.DataInfluencingOn(insecureCreate);


 result = createInfluenced - createInfluenced.DataInfluencingOn(createInfluenced);
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Low_Visibility / Creation_of_Temp_File_With_Insecure_Permissions

Code changes

```
---
+++
@@ -8,4 +8,4 @@
    "File.setReadable", "File.setWritable"});


 CxList insecureWrite = writeToFile - writeToFile.DataInfluencedBy(permissions);
-result = createNewFile.DataInfluencingOn(insecureWrite);
+result = createNewFile.DataInfluencingOn(insecureWrite).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Low_Visibility / Divide_By_Zero

Code changes

```
---
+++
@@ -136,3 +136,4 @@
 }


 result.Add(inputs.DataInfluencingOn(divWithoutIf));
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Low_Visibility / Empty_Password_In_Connection_String

Code changes

```
---
+++
@@ -3,5 +3,5 @@
 CxList password_parameters = All.GetParameters(sql_methods, 2);
 CxList parameters_empty_strings = All.GetParameters(sql_methods.FindByParameterValue(2, "\"\"", BinaryOperator.IdentityEquality), 2);


-result = empty_strings.InfluencingOn(password_parameters);
+result = empty_strings.InfluencingOn(password_parameters).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 result.Add(parameters_empty_strings.FindByType<StringLiteral>());
```

Java / Java_Low_Visibility / Exposure_of_System_Data

Code changes

```
---
+++
@@ -13,4 +13,4 @@
 javaSqlMethods.Add(connections.GetMembersOfTarget().FindByShortName("createStatement"));
 sanitize.Add(javaSqlMethods);


-result = outputs.InfluencedByAndNotSanitized(inputs, sanitize);
+result = outputs.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Low_Visibility / Heap_Inspection

Code changes

```
---
+++
@@ -43,7 +43,7 @@
     "*.guardedstring", "guardedstring",
     "*.KeyStore", "KeyStore",
     "*.SecureString", "SecureString",
-    "JPanel", "JPasswordField", "JScrollPane"};
+    "JPanel", "JPasswordField", "JScrollPane", "JLabel"};
 passwords -= passwords.FindByTypes(safeTypes);
 passwords -= (passwords * Find_Constants());


@@ -89,4 +89,7 @@
 CxList fillSanitizer = methods.FindByMemberAccess("Arrays.fill");
 CxList fillParameters = All.GetParameters(fillSanitizer, 0);
```

```
+CxList safeDecls = result.FindByShortNames(new string[] {"*error*", "*message*", "*hashed*"}, false);
+result -= safeDecls;
+
 result -= result.FindDefinition(fillParameters);
```

Java / Java_Low_Visibility / Information_Exposure_Through_Debug_Log

Code changes

```
---
+++
@@ -13,4 +13,4 @@
 CxList sanitize = Find_Integers();
 sanitize.Add(deadCode);

-result = outputs.InfluencedByAndNotSanitized(inputs, sanitize);
+result = outputs.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Low_Visibility / Information_Exposure_Through_Query_String

Code changes

```
---
+++
@@ -14,5 +14,5 @@
 CxList getMethodsForSensitiveInfo = getMethods.FindByParameters(passwordRelatedNodes);

 result = queryStringGetMethods.DataInfluencingOn(sensitiveInformation).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
-result.Add(passwordRelatedNodes.DataInfluencingOn(queryStringGetMethods),
+result.Add(passwordRelatedNodes.DataInfluencingOn(queryStringGetMethods).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow),
             queryStringMaps.GetMembersOfTarget() * getMethodsForSensitiveInfo);
```

Java / Java_Low_Visibility / Information_Exposure_Through_Server_Log

Code changes

```
---
+++
@@ -11,4 +11,4 @@
 CxList sanitize = Find_Integers();
 sanitize.Add(Find_Dead_Code_Contents());

-result = outputs.InfluencedByAndNotSanitized(inputs, sanitize);
+result = outputs.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Low_Visibility / Information_Leak_Through_Comments

Code changes

```
---
+++
@@ -23,3 +23,4 @@
     {"*.cshtml", "*.xhtml", "*.jsf", "*.jsp", "*.jspf", "*.htm", "*.html", "*.phtml", "*.phtm", "*.rhtml", "*.vm"};
 CxList suspectComments = All.FindByRegexExt(pattern, extensions, true, CxList.CxRegexOptions.SearchOnlyInComments, RegexOptions.IgnoreCase);
 result.Add(suspectComments);
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Low_Visibility / Information_Leak_Through_Persistent_Cookies

Code changes

```
---

+++

@@ -2,4 +2,4 @@


 CxList cookie = All.FindByName("*.setCookies*", false);


-result = cookie.InfluencedBy(psw);

+result = cookie.InfluencedBy(psw).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Low_Visibility / Information_Leak_Through_Shell_Error_Message

Code changes

```
---

+++

@@ -11,4 +11,4 @@

 CxList sanitize = Find_Integers();

 sanitize.Add(Find_Dead_Code_Contents());


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

+result = outputs.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Low_Visibility / Insufficient_Session_Expiration

Code changes

```
---

+++

@@ -36,7 +36,8 @@

 // setMaxInactiveInterval that is influenced by "-1", but with no binary expression in the middle.

 // This will find more results than just looking for "-1" as a parametes, but in (very) extreme

 // cases might give a false positive (for example abs(-1)), but these are really crazy cases

-CxList inJava = maxInactiveInterval.InfluencedByAndNotSanitized(minus1, bin);

+CxList inJava = maxInactiveInterval.InfluencedByAndNotSanitized(minus1, bin)

+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);


 // Add parameters of setMaxInactiveInterval() that are -1 themselves.

 inJava.Add(maxInactiveInterval * minus1.GetFathers().FindByType<UnaryExpr>());
```

Java / Java_Low_Visibility / Log_Forging

Code changes

```
---

+++

@@ -19,7 +19,7 @@

             Find_ObjectCreations().FindByShortName("File*"));


 /* Result */

-result = log.InfluencedByAndNotSanitized(inputs, sanitize);

+result = log.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

 // only get results that intersect with the log output parameters
```

```
 result = result.IntersectWithNodes(All.GetByAncs(logParams));
```

Java / Java_Low_Visibility / Open_Redirect

Code changes

```
---
+++
@@ -4,7 +4,7 @@
 // To remove all File reads
 sanitize.Add(Find_Read_NonDB(), Find_ObjectCreations().FindByShortName("File*"));

-result = redirect.InfluencedByAndNotSanitized(inputs, sanitize);
+result = redirect.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

 // When the flow goes through a "Remote input" (a request to a remote server) it is not vulnerable
 CxList remoteInputs = All.NewCxList(Find_Remote_ReadMethods(), Find_Remote_Connections());
```

Java / Java_Low_Visibility / Portability_Flaw_Locale_Dependent_Comparison

Code changes

```
---
+++
@@ -3,37 +3,44 @@

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

 //Find all parameters used on the Locale sensitve funtions
 CxList localeSensitiveMethodsParams = All.GetParameters(localeSensitiveMethods);

-//Find Locale sensitive functions that use the sanitizer parameter "Locale.*"
-CxList sanitizedMethods = localeSensitiveMethodsParams.FindByName("Locale.*", true)
-                                                      .GetAncOfType<MethodInvokeExpr>();
+//Find Locale sensitive functions that use the sanitizer parameter "Locale.*" or Locale.setDefault(Locale.ROOT);
+CxList sanitizedMethods = All.NewCxList(
+   localeSensitiveMethodsParams.FindByName("Locale.*", true).GetAncOfType<MethodInvokeExpr>(),
+   methods.FindByMemberAccess("Locale", "setDefault"));

 //Remove the sanitized funtions from the list
 localeSensitiveMethods -= sanitizedMethods;

 //Find String search or comparison methods
 CxList targetMethods = methods.FindByShortNames(new string[]{
-       "compareTo*",
-       "contains",
```

```
-       "contentEquals",
-       "endsWith",
-       "equals*",
-       "*indexOf",
-       "matches",
-       "startsWith",
-       "join",
-       "replace"}, false);
+   "compareTo*",
+   "contains",
+   "containsExactly",
+   "contentEquals",
+   "endsWith",
+   "equals*",
+   "*indexOf",
+   "matches",
+   "startsWith",
+   "join",
+   "replace"}, false);


 //Adds all methods from the class Strings - Java and Google Guava API
-targetMethods.Add(methods.FindByMemberAccess("Strings", "*"));
-//Adds all methods from the class StringUtils - Apache Commons Lang3 API
-targetMethods.Add(methods.FindByMemberAccess("StringUtils", "*"));
+targetMethods.Add(
+   methods.FindByMemberAccess("Strings", "*"),
+   methods.FindByMemberAccesses("ObjectUtils", new[] {"notEqual", "equal", "compare"}),
+   methods.FindByMemberAccess("Objects", "equal"),
+   //Adds all methods from the class StringUtils - Apache Commons Lang3 API
+   methods.FindByMemberAccess("StringUtils", "*"));
+
+//Removing specific case
+targetMethods -= methods.FindByShortName("equalsIgnoreCase");


 result = targetMethods.DataInfluencedBy(localeSensitiveMethods);
-//
 result = result.ReduceFlow(CxList.ReduceFlowType.ReduceSmallFlow);
```

Java / Java_Low_Visibility / Reliance_on_DNS_Lookups_in_a_Decision

Code changes

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


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


-result = Find_Conditions().DataInfluencedBy(inetAddress).DataInfluencedBy(ip);
+result = Find_Conditions().DataInfluencedBy(inetAddress).DataInfluencedBy(ip).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Low_Visibility / Use_of_Hard_coded_Security_Constants

Code changes

```
---
+++
@@ -16,4 +16,4 @@

 /// Every problematic parameter influenced (directly) by an ardcoded integer
 result = buffReadParam2 * numbers;
-result.Add(buffReadParam2.InfluencedByAndNotSanitized(numbers, sanitize));
+result.Add(buffReadParam2.InfluencedByAndNotSanitized(numbers, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow));
```

Java / Java_Low_Visibility / Use_of_RSA_Algorithm_without_OAEP

Code changes

```
---
+++
@@ -11,4 +11,4 @@

 CxList notSecured = cipherRSA - paddingOAEP;

-result = cipherInstance.DataInfluencedBy(notSecured);
+result = cipherInstance.DataInfluencedBy(notSecured).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Medium_Threat / CGI_Reflected_XSS_All_Clients

Code changes

```
---
+++
@@ -6,5 +6,5 @@
     CxList sanitized = Find_XSS_Sanitize();
     sanitized.Add(Find_DB_In(), Find_Files_Open());

-    result = inputs.InfluencingOnAndNotSanitized(outputs, sanitized);
+    result = inputs.InfluencingOnAndNotSanitized(outputs, sanitized).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

Java / Java_Medium_Threat / Dangerous_File_Inclusion

Code changes

```
---
+++
@@ -2,4 +2,5 @@
 CxList include = Find_File_Inclusion();
 CxList sanitize = Find_General_Sanitize();

-result = inputs.InfluencingOnAndNotSanitized(include, sanitize, CxList.InfluenceAlgorithmCalculation.NewAlgorithm);
+result = inputs.InfluencingOnAndNotSanitized(include, sanitize, CxList.InfluenceAlgorithmCalculation.NewAlgorithm)
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Medium_Threat / Direct_Use_of_Unsafe_JNI

Code changes

```
---
+++
@@ -48,4 +48,4 @@
 }


 // Parameters that are affected by the input
-result = potentialProblems.DataInfluencedBy(inputs);
+result = potentialProblems.DataInfluencedBy(inputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Medium_Threat / DoS_by_Sleep

Code changes

```
---
+++
@@ -51,4 +51,5 @@
    Find_BinaryExpr().FindByName("=="),
    All.FindAllReferences(Find_FieldDecls().FindByFieldAttributes(Modifiers.Final)));


-result = vulnerableSleepInvokes.InfluencedByAndNotSanitized(Inputs, sanitizers);
+result = vulnerableSleepInvokes.InfluencedByAndNotSanitized(Inputs, sanitizers)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Medium_Threat / Download_of_Code_Without_Integrity_Check

Code changes

```
---
+++
@@ -71,7 +71,7 @@
 CxList sanitizedInputs = unkRefs.FindAllReferences(intersectFirstNodeParam);


 //Possible Tainted Flows
-CxList taintedFlow = outputs.InfluencedBy(inputs);
+CxList taintedFlow = outputs.InfluencedBy(inputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);


 //Flows containing SanitizedInput
 CxList flowWithSanitizedInputs = taintedFlow.IntersectWithNodes(sanitizedInputs);
```

Java / Java_Medium_Threat / External_Control_of_Critical_State_Data

Code changes

```
---
+++
@@ -61,3 +61,4 @@
    permissionsByInput,
    checkByInput,
    controlInfluenceByInput);
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Medium_Threat / External_Control_of_System_or_Config_Setting

Code changes

```
---
+++
@@ -4,4 +4,4 @@
 sanitize.Add(setters.GetTargetOfMembers());
 CxList inputs = Find_Interactive_Inputs();

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

Java / Java_Medium_Threat / Hardcoded_password_in_Connection_String

Code changes

```
---
+++
@@ -81,3 +81,4 @@

 // Add the path from the string/parameter to its method
 result.Add(pswInitMethods.InfluencedByAndNotSanitized(pswStrings, sanitizers));
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Medium_Threat / HttpOnlyCookies

Code changes

```
---
+++
@@ -41,6 +41,6 @@
 unsecuredCookies -= unsecuredCookies.InfluencedBy(getCookies);

 CxList addCookie = headerOutputs.FindByShortNames("addCookie");
-result = addCookie.InfluencedBy(unsecuredCookies);
+result = addCookie.InfluencedBy(unsecuredCookies).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

 result.Add(setHeaders);
```

Java / Java_Medium_Threat / Input_Path_Not_Canonicalized

Code changes

```
---
+++
@@ -28,17 +28,21 @@
     }, false));

 // Find all fileOpens that have some connection with sanitizers
-CxList fileOpenSanitized = All.NewCxList();
-fileOpenSanitized.Add(sanitize,
-   fileOpen.InfluencedBy(sanitize),
-   fileOpen.InfluencingOn(sanitize));
+CxList fileOpenSanitized = All.NewCxList(
+   sanitize,
+   fileOpen.InfluencedBy(sanitize).GetLastNodesInPath(),
+   fileOpen.InfluencingOn(sanitize).GetFirstNodesInPath());
```

```
-// Delete inputs that are sanitized
-inputs -= inputs.InfluencingOn(fileOpenSanitized);
+// Delete outputs that are sanitized
+fileOpen -= fileOpenSanitized;

 // get String.replace(...), String.replaceAll(...) and String.replaceFirst(...) methods
-CxList flowSanitizers = methods.FindByMemberAccess("String.replace*");
-flowSanitizers.Add(integers, methods.FindByMemberAccess("ServletContext.getRealPath"));
+string[] methodNames = new string[]{"String.replace*", "ServletContext.getRealPath"};
+CxList sanitizeMethods = methods.FindByMemberAccesses(methodNames);
+CxList flowSanitizers = All.NewCxList(fileOpenSanitized, integers, sanitizeMethods);

-inputs = inputs.InfluencingOnAndNotSanitized(fileOpen, flowSanitizers);
-result = inputs.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
+result = inputs.InfluencingOnAndNotSanitized(fileOpen, flowSanitizers)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
+
+// Removing duplicated results
+result -= (inputs * fileOpen);
```

Java / Java_Medium_Threat / JSF_CSRF

Code changes

```
---
+++
@@ -38,5 +38,5 @@
    CxList dbWrite = db.DataInfluencedBy(write);
    dbWrite.Add(Find_CSRF_Additional_DB_Write(db));

-   result = inputs.InfluencingOnAndNotSanitized(dbWrite, sanitizedInputs);
+   result = inputs.InfluencingOnAndNotSanitized(dbWrite, sanitizedInputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

Java / Java_Medium_Threat / JWT_Lack_Of_Expiration_Time

Code changes

```
---
+++
@@ -9,4 +9,5 @@
 sanitizedMethod.Add(setClaimsMethods.InfluencedBy(methods.FindByShortName("setExpiration").GetTargetOfMembers()));

 CxList lastNodeFlowJwts = methods.FindByShortName("compact");
-result = lastNodeFlowJwts.InfluencedByAndNotSanitized(jwtsBuildRef, sanitizedMethod);
+result = lastNodeFlowJwts.InfluencedByAndNotSanitized(jwtsBuildRef, sanitizedMethod)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Medium_Threat / JWT_Use_Of_Hardcoded_Secret

Code changes

```
---
+++
@@ -1,26 +1,33 @@
 CxList methods = Find_Methods();
 CxList paramValue = Find_Param().CxSelectDomProperty<Param>(p => p.Value);
-
+
 CxList jwtsRef = methods.FindByMemberAccess("Jwts.builder").GetTargetOfMembers();
 CxList signWithMethod = methods.FindByShortName("signWith");
 CxList refSignWithMethod = signWithMethod.InfluencedBy(jwtsRef);
-
+
 CxList signWithMet = refSignWithMethod.FindByShortName("signWith");
 CxList signWithParam = paramValue.GetParameters(signWithMet);
 CxList parameterInMethod = signWithParam.FindByType("Key");
-
-CxList flows = parameterInMethod.DataInfluencedBy(Find_Strings());
-
+
+CxList flows = Find_Strings().DataInfluencingOn(parameterInMethod);
+
 flows.Add(signWithParam.FindByType<StringLiteral>());
-
+
 CxList attributesValue = Create_Flow_Spring_CustomAttribute();
-
+
 // Sanitize flows were JWT is not being signed through signWith method
 attributesValue -= attributesValue.SanitizeCxList(paramValue.GetParameters(signWithMet, 1)
     .Contained(attributesValue, CxList.GetStartEndNodesType.AllNodes));
-
+
 CxList sanitizedMethods = methods.FindByMemberAccess("Cipher.getInstance");
 attributesValue -= attributesValue.IntersectWithNodes(sanitizedMethods);
-
+
 flows.Add(attributesValue);
+
+CxList getKeyMethods = methods.FindByMemberAccess("KeyStore.getKey");
+CxList getKeyFlows = flows.IntersectWithNodes(getKeyMethods);
+CxList validParams = paramValue.GetParameters(getKeyMethods, 1);
+CxList invalidGetKeyFlows = getKeyFlows - getKeyFlows.IntersectWithNodes(validParams);
+flows = flows.SanitizeCxList(invalidGetKeyFlows);
+
 result = flows.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Medium_Threat / Plaintext_Storage_of_a_Password

Code changes

```
---
+++
@@ -20,4 +20,4 @@
 CxList sanitize = Find_General_Sanitize();


 // All passwords of getConnection that are affected by a non-interactive input, and not well sanitized
-result = inputs.InfluencingOnAndNotSanitized(password, sanitize);
+result = inputs.InfluencingOnAndNotSanitized(password, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Medium_Threat / Privacy_Violation

Code changes

```
---
+++
@@ -1,56 +1,7 @@
 // This query searches for variables and constants that could contain personal sensitive data,
 // which is streamed to an output.
-CxList strings = Find_Strings();
-CxList integerLiteral = Find_IntegerLiterals();
-CxList literals = All.NewCxList(strings, integerLiteral);
-CxList nullLiteral = Find_NullLiteral();
-CxList typeRefs = Find_TypeRef();
-CxList unknRefs = Find_UnknownReference();
-CxList methods = Find_Methods();
-CxList constructors = Find_ConstructorDecl();
-// 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(Find_Personal_Info(), Find_Password_Info());
-personal_info -= strings;
-
-// 1) Exclude variables that are all uppercase
-//       Usually describes the pattern of the data, such as PASSWORDPATTERN, PASSWORDTYPE, ...
-personal_info -= personal_info.FilterByDomProperty<CSharpGraph>(pi => !pi.ShortName.Any(c => char.IsLower(c)));
-
-// 2) Exclude constants that are assigned a literal
-CxList constants = personal_info * Find_Constants();
-CxList allConstRef = personal_info.FindAllReferences(constants);
-CxList allConstRefOrigin = All.NewCxList(allConstRef);
-
-// Find all assignments of string or integer literals
-CxList ConstAssignedL = literals.FindByFathers(allConstRef.FindByType<Declarator>());
-
-// Remove assignments of constants to string or integer literals
-allConstRef -= personal_info.FindAllReferences(ConstAssignedL.GetFathers());
-
-// Remove assignments of constants to null literals
-CxList declWithNull = allConstRef * nullLiteral.GetFathers().FindByType<Declarator>();
-
```

```
-// Constants are assigned null value by default if the real assignment is not in the declaration line,
-// and so the implicit assignment to null is irrelevant.
-// e.g. final x; is parsed as final x = null; although x can be assigned later
-CxList allRefs = allConstRef.FindAllReferences(declWithNull);
-
-// FindByAssignmentSide(left) finds the real assignments, eg. x = ...
-CxList toRemove = declWithNull - allRefs.FindDefinition(allRefs.FindByAssignmentSide(CxList.AssignmentSide.Left));
-allConstRef -= toRemove;
-
-// Find all assignments to constants
-CxList constAssignments = allConstRef.FindByAssignmentSide(CxList.AssignmentSide.Left).GetFathers();
-
-// Find assignments of literals to constant personal_info and remove them from results
-CxList PI_literal = literals.GetFathers() * constAssignments;
-allConstRef -= allConstRef.FindAllReferences(allConstRef.FindByFathers(PI_literal));
-
-// Remove from personal_info all references that were removed above
-personal_info -= (allConstRefOrigin - allConstRef);
-
+// 1) Calculate inputs.
 CxList inputs = All.NewCxList(
    Find_DB_Out(),
    Find_Environment_Inputs(),
@@ -59,87 +10,101 @@
    Find_Local_Console_Inputs(),
    Find_Portlets_Inputs());

-// We must find responses with sensitive types
-//Get classes with sensitive fields
-CxList sensitiveClass = personal_info.GetAncOfType<ClassDecl>();
-sensitiveClass.Add(typeRefs.FindByType(sensitiveClass).GetAncOfType<ClassDecl>());
-//Get unknown references with sensitive type
-CxList sensitiveTypeVars = typeRefs.FindByType(sensitiveClass).GetAncOfType<VariableDeclStmt>();
-CxList sensitiveTypeDecls = Find_Declarators().GetByAncs(sensitiveTypeVars);
-CxList sensitiveTypeUsage = unknRefs.FindAllReferences(sensitiveTypeDecls);
-CxList sensitiveResponse = sensitiveTypeUsage * Find_HTTP_Responses();
+// If there are no inputs, we don't need to calculate anything else in this query.
+if (inputs.Count == 0)
+   return All.NewCxList();

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

-// 3) Add exceptions (that could be thrown) to outputs.
```

```
-CxList exceptions = Find_ObjectCreations().FindByName("*Exception");
-CxList exceptionsCtors = constructors.FindByName("*Exception");
+// 2) Calculate outputs.
+// Add exceptions (that could be thrown) to the outputs.
+CxList exceptionCreations = Find_ObjectCreations().FindByName("*Exception");

-// Handle the case where the super (base) constructor of the exception is used to create a new throwable exception
-CxList exceptionsCtorsWithSuper = exceptionsCtors.FilterByDomProperty<ConstructorDecl>(c => c.BaseParameters.Count > 0);
+// Handle the case where the super (base) constructor of the exception is used to create a new throwable exception.
+CxList exceptionCtorsWithSuper = Find_ConstructorDecl().FindByName("*Exception")
+   .FilterByDomProperty<ConstructorDecl>(c => c.BaseParameters.Count > 0);

-// Define outputs
+CxList exceptionOutputs = All.NewCxList(exceptionCreations, exceptionCtorsWithSuper);
+
 CxList outputs = All.NewCxList(
    Find_Outputs(),
-   exceptions,
-   exceptionsCtorsWithSuper,
-   Find_Cloud_Outputs()
-   );
+   Find_Cloud_Outputs(),
+   exceptionOutputs);

-// Define sanitize
-CxList sanitize = All.NewCxList(Find_DB(), Find_Encrypt(), Find_UnitTest_Code(), Find_HashSanitize(), Find_DataSource_Sanitizers());
+// If there are no outputs, we don't need to calculate anything else in this query.
+if (outputs.Count == 0)
+   return All.NewCxList();

-// Add additional "integer" sanitizers
-sanitize.Add(All.FindByShortNames(new string[] {"size", "length", "Index*", "indexOf"}, false),
-   All.FindByName("*boolean.class.cast", StringComparison.OrdinalIgnoreCase),
-   methods.FindByMemberAccess("Boolean.parse*"), methods.FindByReturnType("bool"), Find_BooleanLiteral(),
-   methods.FindByMemberAccess("Process.waitFor"));

-CxList javaSqlMethods = methods.FindByMemberAccess("DriverManager.getConnection");
-CxList connections = unknRefs.FindAllReferences(javaSqlMethods.GetAssignee());
-javaSqlMethods.Add(connections.GetMembersOfTarget().FindByShortName("createStatement"));
-sanitize.Add(javaSqlMethods);
+// 3) Calculate sanitizers.
+CxList methods = Find_Methods();
+CxList booleanRelated = All.NewCxList(
+   Find_BooleanLiteral(),
+   methods.FindByMemberAccess("Boolean.parse*"),
+   methods.FindByReturnType("boolean"));

-sanitize.Add(All.FindByShortNames(new string[]{"*clientname*", "*username*"}, false));
```

```
+string[] memberNames = new string[] {"size", "length", "Index*", "indexOf", "*clientname*", "*username*"};
+CxList possibleMembers = All.NewCxList(Find_MemberAccesses(), methods);
+CxList sanitizeMembers = possibleMembers.FindByShortNames(memberNames, false);

-// Split personal_info into variables and constants
-CxList variableRef = personal_info - allConstRef;
+CxList unknownRefs = Find_UnknownReference();
+CxList sqlMethods = methods.FindByMemberAccess("DriverManager.getConnection");
+CxList connections = unknownRefs.FindAllReferences(sqlMethods.GetAssignee());
+sqlMethods.Add(connections.GetMembersOfTarget().FindByShortName("createStatement"));

-// find declarators of constants and variables so they can be removed
-// declarators are not a part of the flow from input to output
-// eg. string x = ___ is parsed as: (Declarator) string (UnknownReference) x (AssignExpr) = (value / expression / literal)___
-// the real flow is from the UnknownReference and not the Declarator
-CxList declarator = personal_info.FindByType<Declarator>();
+CxList waitFor = methods.FindByMemberAccess("Process.waitFor");

-// remove the declaration from the references of the variables and constants
-variableRef -= declarator;
-allConstRef -= declarator;
+CxList sanitize = All.NewCxList(
+   Find_DataSource_Sanitizers(),
+   Find_DB(),
+   Find_Encrypt(),
+   Find_HashSanitize(),
+   Find_UnitTest_Code(),
+   booleanRelated,
+   sanitizeMembers,
+   sqlMethods,
+   waitFor);

-// remove duplicate results from object creations
-CxList variableRefsConstructs = variableRef.GetByAncs(constructors);
-variableRef -= variableRefsConstructs.InfluencedBy(variableRefsConstructs.FindByType<ParamDecl>());

-// Find all constants that are assigned from an input (directly or indirectly) and are influencing an output
-CxList constInfluencedByInputAux = inputs.InfluencingOnAndNotSanitized(outputs, sanitize);
-CxList constInfluencedByInput = All.NewCxList();
-foreach (CxList elem in constInfluencedByInputAux.GetCxListByPath())
-{
-   if ((elem.GetStartAndEndNodes(CxList.GetStartEndNodesType.AllNodes) * allConstRef).Count > 0)
-   {
-       constInfluencedByInput.Add(elem);
-   }
-}
+// 4) Calculate personal information.
+// eg. String password / Integer ssn
```

```
+CxList personalInfo = All.NewCxList(Find_Personal_Info(), Find_Password_Info());


-// find all variables that are influencing an output
-CxList variableRefPath = outputs.InfluencedByAndNotSanitized(variableRef, sanitize);
-variableRefPath.Add(variableRef * outputs - sanitize);
+CxList constants = personalInfo * Find_ConstantDecl();
+CxList constantRefs = personalInfo.FindAllReferences(constants);


-result = variableRefPath;
-result.Add(constInfluencedByInput);
+CxList sanitizeAssignees = personalInfo * sanitize.GetAssignee();
+CxList sanitizeRefs = personalInfo.FindAllReferences(sanitizeAssignees);


-result = result.ReduceFlow(CxList.ReduceFlowType.ReduceSmallFlow);
+// Exclude variables that are all uppercase.
+// Usually describes the pattern of the data such as PASSWORDPATTERN, PASSWORDTYPE, ...
+CxList uppercaseInfo = personalInfo.Filter(pi => !pi.ShortName.Any(c => char.IsLower(c)));
+
+CxList exclusions = All.NewCxList(constantRefs, sanitizeRefs, uppercaseInfo);
+CxList relevantTypes = All.NewCxList(Find_ParamDecl(), possibleMembers, unknownRefs);
+
+personalInfo *= relevantTypes - exclusions;
+
+// Find also responses with sensitive types.
+CxList typeRefs = Find_TypeRef();
+CxList sensitiveClasses = personalInfo.GetAncOfType<ClassDecl>();
+sensitiveClasses.Add(typeRefs.FindByType(sensitiveClasses).GetAncOfType<ClassDecl>());
+
+// Get unknown references with a sensitive type.
+CxList sensitiveTypeVars = typeRefs.FindByType(sensitiveClasses).GetAncOfType<VariableDeclStmt>();
+CxList sensitiveTypeDecls = Find_Declarators().GetByAncs(sensitiveTypeVars);
+CxList sensitiveTypeUsages = unknownRefs.FindAllReferences(sensitiveTypeDecls);
+CxList sensitiveResponses = sensitiveTypeUsages * Find_HTTP_Responses();
+
+
+// 5) Calculate the flows from inputs to outputs through personal information.
+CxList taintedPersonalInfo = All.NewCxList(
+   inputs.InfluencingOn(personalInfo).GetLastNodesInPath(),
+   personalInfo * inputs,
+   sensitiveResponses);
+
+// Find all tainted personal information that is influencing an output.
+CxList flows = taintedPersonalInfo.InfluencingOnAndNotSanitized(outputs, sanitize)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceSmallFlow);
+
+taintedPersonalInfo -= flows.GetLastNodesInPath();
+result = All.NewCxList(flows, taintedPersonalInfo * outputs);
```

Java / Java_Medium_Threat / ReDoS_In_Pattern

Code changes

```
---
+++
@@ -17,4 +17,4 @@
 CxList matchSplit = All.NewCxList(match, split);


 // Find relevant matches
-result = activeEvilRegexes.DataInfluencingOn(matchSplit);
+result = activeEvilRegexes.DataInfluencingOn(matchSplit).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Medium_Threat / Reliance_on_Cookies_without_Validation

Code changes

```
---
+++
@@ -57,3 +57,4 @@


 /// The final result - all the relevant cases
 result.Add(permissionsByCookies, checkByCookies, controlInfluenceByCookies);
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Medium_Threat / SSL_Verification_Bypass

Code changes

```
---
+++
@@ -86,7 +86,9 @@
 //Add any certs that belong to a class whose checkServerTrusted method can actually return an exception
 CxList liveThrows = throwStmtList - Find_Dead_Code_AbsInt().FindByType<ThrowStmt>();
 CxList checksThatThrow = liveThrows.GetAncOfType<MethodDecl>().FindByShortName("checkServerTrusted");
-safeCerts.Add(newCerts.GetByAncs(checksThatThrow.GetAncOfType<ClassDecl>()));
+safeCerts.Add(newCerts.GetByAncs(checksThatThrow.GetAncOfType<ClassDecl>()),
+   // Remove results that do not involve any sort of configuration they are only assignments
+   Find_Declarators().GetByAncs(newCerts.GetAncOfType<VariableDeclStmt>()));


 CxList goodCerts = newCerts - safeCerts;


@@ -123,4 +125,4 @@


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


-result = result.NotInfluencedBy(validateMethods).NotInfluencingOn(validateMethods);
+result = result.NotInfluencedBy(validateMethods).NotInfluencingOn(validateMethods).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Medium_Threat / SSRF

Code changes

```
---
+++
@@ -1,49 +1,15 @@
```

```
-CxList methods = Find_Methods();
-CxList unkRefs = Find_UnknownReference();
-CxList paramValue = Find_Param().CxSelectDomProperty<Param>(p => p.Value);
+CxList methodDecls = Find_MethodDecls();

 CxList inputs = All.NewCxList(Find_Interactive_Inputs(), Find_Queue_Inputs());
-
 // Remove argc/argv from inputs
-CxList mainDeclarations = All.FindByShortName("*main*").FindByType<MethodDecl>();
-CxList argsInputs = paramValue.GetParameters(mainDeclarations);
+CxList mainDeclarations = methodDecls.FindByShortName("*main*");
+CxList argsInputs = Find_Param().GetParameters(mainDeclarations).CxSelectDomProperty<Param>(p => p.Value);
 inputs -= argsInputs;

 CxList sanitizers = Find_Remote_Requests_Sanitize();

-CxList requests = Find_Remote_Requests();
-
-//Transport.send must be influenced by a Properties object with key "mail.smtp.host" influenced by user input
-CxList sendMethods = methods.FindByMemberAccess("Transport.send");
-CxList relevantKey = Find_Strings().FindByShortName("mail.smtp.host");
-CxList propertySetMethods = methods.FindByMemberAccesses("Properties",
-   new[]{"setProperty", "put", "putAll", "putIfAbsent"});
-
-IEnumerable<IAbstractValue> keyAbsValue = relevantKey.CxSelectElementValue<Expression, IAbstractValue>(e => e.AbsValue);
-CxList relevantKeyProperties = paramValue.GetParameters(propertySetMethods).FindByAbstractValue(
-   absValue => absValue.IncludedIn(keyAbsValue.FirstOrDefault()));
-
-requests -= paramValue.GetParameters(sendMethods.NotInfluencedBy(relevantKeyProperties));
-
-// remove 2nd and 3rd params from DriverManager.getConnection since only the first one can be valid
-CxList driverManagerGetConnections = methods.FindByMemberAccess("DriverManager.getConnection");
-requests -= requests.GetParameters(driverManagerGetConnections, 1);
-requests -= requests.GetParameters(driverManagerGetConnections, 2);
+CxList requests = Find_Remote_Connections_Destinations();

 result = requests.InfluencedByAndNotSanitized(inputs, sanitizers)
     .ReduceFlowByPragma()
     .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
-
-//Removing openStream results on previously Opened Connection
-CxList resultLastNodes = result.GetLastNodesInPath();
-CxList openConnection = resultLastNodes.FindByShortName("openConnection");
-CxList openStream = resultLastNodes.FindByShortName("openStream");
-CxList toRemove = All.NewCxList();
-foreach(CxList opConn in openConnection){
-   CxList openConnectionVar = opConn.GetTargetOfMembers();
-   CxList subsequentVarRefs = unkRefs.FindAllReferences(openConnectionVar)
```

```
-       .Filter(x => x.Line > openConnectionVar.GetDOMPropertiesOfFirst().Line);
-   CxList openStreamSameVar = subsequentVarRefs.GetMembersOfTarget() * openStream;
-   toRemove.Add(openStreamSameVar.InfluencedByAndNotSanitized(inputs, sanitizers));
-}
-result -= toRemove;
```

Java / Java_Medium_Threat / Unvalidated_Forwards

Code changes

```
---
+++
@@ -12,4 +12,4 @@
    unkRefStrMethd.GetParameters(unkRefStrMethd.FindByShortName("forward")));


 result = inputs.InfluencingOnAndNotSanitized(requestDispatcherParams, sanitizers)
-   .InfluencingOnAndNotSanitized(unvalidatedForwards, sanitizers);
+   .InfluencingOnAndNotSanitized(unvalidatedForwards, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Medium_Threat / Use_of_Cryptographically_Weak_PRNG

Code changes

```
---
+++
@@ -3,4 +3,4 @@
 CxList outputs = Find_Encrypt();
 outputs.Add(Find_HashSanitize());

-result = inputs.DataInfluencingOn(outputs);
+result = inputs.DataInfluencingOn(outputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Medium_Threat / Use_of_Native_Language

Code changes

```
---
+++
@@ -22,4 +22,4 @@
 externalMethodParams -= externalMethodParams.GetByAncs(paramsIfs.GetAncOfType<IfStmt>());


 // Parameters that are affected by the input
-result = externalMethodParams.DataInfluencedBy(inputs);
+result = externalMethodParams.DataInfluencedBy(inputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Medium_Threat / XQuery_Injection

Code changes

```
---
+++
@@ -14,4 +14,4 @@
 // Jaxp and Saxon
 candidates.Add(Find_XQuery_Injection_Candidates());


-result = candidates.InfluencedByAndNotSanitized(inputs, sanitize);
```

```
+result = candidates.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Spring / Spring_Comparison_Timing_Attack

Code changes

```
---
+++
@@ -1,4 +1,4 @@
 CxList methods = Find_Methods();
 CxList equalsMethod = methods.FindByShortName("equals");
 CxList insecurePasswordEncoder = methods.FindByMemberAccess("*PasswordEncoder.encode");
-result = insecurePasswordEncoder.DataInfluencingOn(equalsMethod);
+result = insecurePasswordEncoder.DataInfluencingOn(equalsMethod).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Spring / Spring_ModelView_Injection

Code changes

```
---
+++
@@ -6,4 +6,4 @@
 ModelAndView = All.GetParameters(ModelAndView, 0);
 CxList sanitize = Find_General_Sanitize();
 sanitize.Add(Find_General_Sanitize_Injection());
-result = ModelAndView.InfluencedByAndNotSanitized(inputs, sanitize);
+result = ModelAndView.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Java / Java_Spring / Spring_Overly_Permissive_Cross_Origin_Resource_Sharing_Policy

Code changes

```
---
+++
@@ -81,10 +81,6 @@
                     {
                         hasPermissiveCorsPolicy.Add(corsSpecificPolicy.ConcatenatePath(mappingMethod));
                     }
-                    else
-                    {
-                        hasPermissiveCorsPolicy.Add(mappingMethod);
-                    }
                 }
             }
         }
```

Java / Java_Stored / Stored_Open_Redirect

Code changes

```
---
+++
@@ -1,13 +1,6 @@
-CxList readNonDB = Find_Read_NonDB();
-// System.getProperties should not be considered as input
-CxList listSystemGetPropertiesInInputs = readNonDB.FindByMemberAccesses(new string [] {
```

```
-   "System.getProperty","System.getProperties","System.getenv"});
+CxList inputs = All.NewCxList(Find_Read_NonDB(), Find_DB_Out());

-CxList inputs = readNonDB - listSystemGetPropertiesInInputs;
-inputs.Add(Find_DB_Out());
-
-CxList sanitize = Find_Integers();
-sanitize.Add(Find_HashSanitize());
+CxList sanitize = All.NewCxList(Find_Integers(), Find_HashSanitize());

 CxList redirect = Find_Redirects();
```

Java / Java_Stored / Stored_XPath_Injection

Code changes

```
---
+++
@@ -1,10 +1,4 @@
-CxList readNonDB = Find_Read_NonDB();
-// System.getProperties should not be considered as input
-CxList listSystemGetPropertiesInInputs = readNonDB.FindByMemberAccesses(new string [] {
-   "System.getProperty","System.getProperties","System.getenv"});
-
-CxList inputs = readNonDB - listSystemGetPropertiesInInputs;
-inputs.Add(Find_DB_Out());
+CxList inputs = All.NewCxList(Find_Read_NonDB(), Find_DB_Out());

 CxList sanitized = Find_XPath_Sanitize();
```

JavaScript / JavaScript_Angular / Angular_Client_DOM_XSS

Code changes

```
---
+++
@@ -6,5 +6,5 @@
     CxList outputs = Angular_Find_Outputs_XSS();
     CxList desanitizer = methods.FindByShortName("bypassSecurityTrust*");
     CxList possibleXSS = outputs.DataInfluencedBy(inputs);
-    result = possibleXSS.IntersectWithNodes(desanitizer);
+    result = possibleXSS.IntersectWithNodes(desanitizer).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

JavaScript / JavaScript_Angular / Angular_Client_Stored_DOM_XSS

Code changes

```
---
+++
@@ -6,5 +6,5 @@
     CxList desanitizer = methods.FindByShortName("bypassSecurityTrust*");
```

```
    CxList possibleXSS = outputs.DataInfluencedBy(inputs);
-   result = possibleXSS.IntersectWithNodes(desanitizer);
+   result = possibleXSS.IntersectWithNodes(desanitizer).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

JavaScript / JavaScript_AWS_Lambda / DynamoDB_NoSQL_Injection

Code changes

```
---
+++
@@ -5,4 +5,4 @@
 CxList inputs = Find_Inputs();
 CxList sanitize = DynamoDB_Find_NoSQL_Injection_Sanitize();

-result = outputs.InfluencedByAndNotSanitized(inputs, sanitize);
+result = outputs.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_AWS_Lambda / Unrestricted_Read_S3

Code changes

```
---
+++
@@ -23,3 +23,5 @@
 result = inputs.InfluencingOnAndNotSanitized(awsOutputs, sanitizers)
     .IntersectWithNodes(s3Commands)
     .IntersectWithNodes(keyDecls);
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_AWS_Lambda / Unrestricted_Write_S3

Code changes

```
---
+++
@@ -29,3 +29,5 @@
 CxList inputs = Find_Inputs();
 result = inputs.InfluencingOnAndNotSanitized(s3DbIn, sanitizers)
     .IntersectWithNodes(keyDecls);
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_AWS_Lambda / User_Based_SDK_Configurations

Code changes

```
---
+++
@@ -18,7 +18,9 @@

     //AWS.config.update first parameter
     CxList updateFirstParam = All.GetParameters(configApi.FindByShortName("update"), 0);
-    result.Add(inputs.InfluencingOnAndNotSanitized(updateFirstParam, sanitizers));
```

```
+   result.Add(inputs.InfluencingOnAndNotSanitized(updateFirstParam, sanitizers)
+       .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow)
+   );


    //AWS.config.loadFromPath
    result.Add(configApi.FindByShortName("loadFromPath"));
```

JavaScript / JavaScript_Cordova / Cordova_Code_Injection

Code changes

```
---
+++
@@ -5,4 +5,4 @@
 CxList code = PhoneGap_Find_Code_Injection();
 CxList sanitize = basic_Sanitize();

-result = code.InfluencedByAndNotSanitized(inputs, sanitize);
+result = code.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Cordova / Cordova_File_Disclosure

Code changes

```
---
+++
@@ -5,4 +5,4 @@
 CxList inputs = Find_Inputs();
 CxList sanitize = basic_Sanitize();

-result = reads.InfluencedByAndNotSanitized(inputs, sanitize);
+result = reads.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Cordova / Cordova_File_Manipulation

Code changes

```
---
+++
@@ -5,4 +5,4 @@
 CxList inputs = Find_Inputs();
 CxList sanitize = basic_Sanitize();

-result = write.InfluencedByAndNotSanitized(inputs, sanitize);
+result = write.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Cordova / Cordova_Open_Redirect

Code changes

```
---
+++
@@ -5,4 +5,4 @@
 CxList redir = PhoneGap_Find_Open_Redirect();
 CxList sanitize = basic_Sanitize();
```

```
-result = redir.InfluencedByAndNotSanitized(inputs, sanitize);
+result = redir.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_High_Risk / Client_DOM_Stored_XSS

Code changes

```
---
+++
@@ -10,6 +10,10 @@
    Find_Cookie(),
    Find_DB_Out(),
    Find_XHR_Response());
+
+//Remove sources from error handlers
+CxList fieldError = Find_FieldDecls().FindByShortName("error").GetByAncs(Find_XHR_Wrappers());
+sources -= sources.GetByAncs(fieldError);

 //All sources and elements influenced by them, that have a sanitizer in their ancestors, are also sanitizers
 CxList sourcesAndInfluencedBy = All.NewCxList(
```

JavaScript / JavaScript_High_Risk / Client_Dynamic_File_Inclusion

Code changes

```
---
+++
@@ -8,4 +8,4 @@
 CxList requires = Find_Methods().FindByShortNames(new string[]{"require", "requirejs", "append*"});

 sanitizers.Add(unkRefs.FindAllReferences(varsInIfConditions));
-result = inputs.InfluencingOnAndNotSanitized(requires, sanitizers);
+result = inputs.InfluencingOnAndNotSanitized(requires, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_High_Risk / Client_Resource_Injection

Code changes

```
---
+++
@@ -34,4 +34,4 @@

 CxList sanitize = inputs.DataInfluencingOn(finalStartsWithMethods);

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

JavaScript / JavaScript_High_Risk / Prototype_Pollution

Code changes

```
---
+++
@@ -7,6 +7,10 @@
 CxList indexerRefs = Find_IndexerRefs();
 CxList methods = Find_Methods();
```

```
 CxList strings = Find_Strings();
+CxList unkRefs = Find_UnknownReference();
+CxList constants = Find_Constants();
+CxList declarators = Find_Declarators();
+
 // inputs
 CxList inputs = All.NewCxList(Find_Inputs(), Find_Cloud_Interactive_Inputs());


@@ -68,4 +72,15 @@


 sanitizers.Add(All.GetParameters(ajvValidateMethods));


+
+CxList relevIfs = All.NewCxList();
+CxList constantIfs = unkRefs.FindAllReferences(declarators.GetByAncs(constants)).GetAncOfType<IfStmt>();
+CxList includesMethods = methods.FindByShortName("includes", false);
+relevIfs.Add(includesMethods.GetByAncs(constantIfs).GetAncOfType<IfStmt>());
+relevIfs.Add(unkRefs.FindByType("RegExp").GetAncOfType<IfStmt>());
+CxList condRefs = unkRefs.FindAllReferences(unkRefs.GetByAncs(relevIfs.CxSelectDomProperty<IfStmt>(x => x.Condition)));
+CxList relevRefs = unkRefs.GetByAncs(relevIfs.CxSelectDomProperty<IfStmt>(
+   x => x.Condition.ShortName == "Not" ? x.FalseStatements : x.TrueStatements));
+sanitizers.Add(condRefs * relevRefs);
+
 result = inputs.InfluencingOnAndNotSanitized(outputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / Javascript_Kony / Kony_Code_Injection

Code changes

```
---
+++
@@ -8,6 +8,7 @@
    CxList sanitize = Code_Injection_Sanitize();


    result.Add(Eval.FindByType<Param>() * inputs - sanitize,
-           (Eval).InfluencedByAndNotSanitized(inputs, sanitize, CxList.InfluenceAlgorithmCalculation.NewAlgorithm),
+           (Eval).InfluencedByAndNotSanitized(inputs, sanitize, CxList.InfluenceAlgorithmCalculation.NewAlgorithm)
+               .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow),
            Find_Source_Equals_Sink(inputs, Eval));
 }
```

JavaScript / Javascript_Kony / Kony_Hardcoded_EncryptionKey

Code changes

```
---
+++
@@ -14,5 +14,5 @@
    CxList sinks = cryptoKey_Params.Clone();
    CxList sanitizers = invokes.FindByName("kony.crypto.newKey");


-   result = sources.InfluencingOnAndNotSanitized(sinks, sanitizers);
```

```
+   result = sources.InfluencingOnAndNotSanitized(sinks, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

JavaScript / Javascript_Kony / Kony_Path_Injection

Code changes

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

    CxList sanitizers = basic_Sanitize();

-   result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers);
+   result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

JavaScript / Javascript_Kony / Kony_Second_Order_SQL_Injection

Code changes

```
---
+++
@@ -11,5 +11,5 @@
    CxList outputs = Kony_DB_In();
    CxList sanitizers = Find_Kony_Sql_Sanitize();

-   result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers);
+   result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

JavaScript / Javascript_Kony / Kony_SQL_Injection

Code changes

```
---
+++
@@ -7,5 +7,5 @@
    CxList outputs = Kony_DB_In();
    CxList sanitizers = Find_Kony_Sql_Sanitize();

-   result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers);
+   result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

JavaScript / Javascript_Kony / Kony_Stored_Code_Injection

Code changes

```
---
+++
@@ -13,6 +13,7 @@
    CxList sanitize = Code_Injection_Sanitize();

    result.Add(Eval.FindByType<Param>() * inputs - sanitize,
-              Eval.InfluencedByAndNotSanitized(inputs, sanitize, CxList.InfluenceAlgorithmCalculation.NewAlgorithm),
```

```
+               Eval.InfluencedByAndNotSanitized(inputs, sanitize, CxList.InfluenceAlgorithmCalculation.NewAlgorithm)
+               .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow),
                Find_Source_Equals_Sink(inputs, Eval));
 }
```

JavaScript / Javascript_Kony / Kony_URL_Injection

Code changes

```
---
+++
@@ -7,5 +7,5 @@
    CxList outputs = Kony_HTTP_Outputs();
    CxList sanitizers = basic_Sanitize();

-   result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers);
+   result = outputs.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

JavaScript / Javascript_Kony / Kony_Use_WeakEncryption

Code changes

```
---
+++
@@ -13,5 +13,5 @@
    List<string> weakAlgo_Names = new List<string>{"tripledes"};
    CxList weakAlgo = Find_String_Literal().FindByShortNames(weakAlgo_Names);

-   result = weakAlgo.DataInfluencingOn(algoParam);
+   result = weakAlgo.DataInfluencingOn(algoParam).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

JavaScript / Javascript_Kony / Kony_Use_WeakHash

Code changes

```
---
+++
@@ -13,5 +13,5 @@
    List<string> weakAlgo_Names = new List<string>{"sha1", "md5"};
    CxList weakAlgo = Find_String_Literal().FindByShortNames(weakAlgo_Names);

-   result = weakAlgo.DataInfluencingOn(algoParam);
+   result = weakAlgo.DataInfluencingOn(algoParam).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

JavaScript / Javascript_Lightning / Lightning_DOM_XSS

Code changes

```
---
+++
@@ -9,5 +9,5 @@
    sanitize.Add(Find_XSS_Sanitize(),
                 Lightning_XSS_Sanitizer());
```

```
-   result = inputs.InfluencingOnAndNotSanitized(outputs, sanitize);
+   result = inputs.InfluencingOnAndNotSanitized(outputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

JavaScript / Javascript_Lightning / Lightning_Stored_XSS

Code changes

```
---
+++
@@ -4,5 +4,5 @@
    CxList sanitize = basic_Sanitize();
    sanitize.Add(Find_XSS_Sanitize(),
                 Lightning_XSS_Sanitizer());
-   result = outputs.InfluencedByAndNotSanitized(dbOut, sanitize);
+   result = outputs.InfluencedByAndNotSanitized(dbOut, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

JavaScript / JavaScript_Low_Visibility / Client_Cookies_Inspection

Code changes

```
---
+++
@@ -6,4 +6,4 @@
 CxList cookies = Find_Cookie();
 CxList personal = Find_Personal_Info();

-result = cookies.DataInfluencedBy(personal);
+result = cookies.DataInfluencedBy(personal).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Low_Visibility / Client_Empty_Password

Code changes

```
---
+++
@@ -3,4 +3,4 @@
 CxList passwords = Find_String_Literal().GetParameters(open, 4);
 CxList relevantPwd = passwords.FindByShortName("");

-result = open.DataInfluencedBy(relevantPwd);
+result = open.DataInfluencedBy(relevantPwd).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Low_Visibility / Client_HTML5_Easy_To_Guess_Database_Name

Code changes

```
---
+++
@@ -42,4 +42,4 @@
 CxList relevantMethods = methods.FindByShortName("openDatabase");
 relevantMethods.Add(indexedDBOpen);

-result = relevantMethods.InfluencedBy(relevantStrings);
```

```
+result = relevantMethods.InfluencedBy(relevantStrings).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Low_Visibility / Client_Remote_File_Inclusion

Code changes

```
---
+++
@@ -76,4 +76,5 @@
 CxList appendChildList = documentDom.GetRightmostMember();
 appendChildList.Add(appendChildList.GetFathers().FindByType<IndexerRef>().GetRightmostMember());

-result = appendChildList.FindByParameters(paramToCatch).InfluencedBy(createElmentImp);
+result = appendChildList.FindByParameters(paramToCatch).InfluencedBy(createElmentImp)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Low_Visibility / Client_Weak_Password_Authentication

Code changes

```
---
+++
@@ -1,4 +1,4 @@
 CxList open = Find_XmlHttp_Open();
 result = Find_String_Literal().GetParameters(open, 4);

-result = open.DataInfluencedBy(result);
+result = open.DataInfluencedBy(result).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Low_Visibility / Information_Exposure_Through_Query_Strings

Code changes

```
---
+++
@@ -11,3 +11,5 @@
    flow.Add(passwordParam.GetByAncs(urls.GetAncOfType<AssignExpr>()));
    result.Add(getJson.InfluencedBy(flow));
 }
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Low_Visibility / Insufficiently_Protected_Credentials

Code changes

```
---
+++
@@ -11,3 +11,5 @@
    flow.Add(passwordParam.GetByAncs(urls.GetAncOfType<AssignExpr>()));
    result.Add(getJson.InfluencedBy(flow));
 }
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Medium_Threat / Client_DOM_Cookie_Poisoning

Code changes

```
---
+++
@@ -1,4 +1,4 @@
 CxList cookies = Find_Cookie();
 CxList url = Find_Inputs();

-result = cookies.DataInfluencedBy(url);
+result = cookies.DataInfluencedBy(url).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Medium_Threat / Client_HTML5_Information_Exposure

Code changes

```
---
+++
@@ -18,4 +18,4 @@
 }
 CxList temp = WebMessagingOutputOrigin * star;
 temp.Add(WebMessagingOutputOrigin.DataInfluencedBy(star));
-result = WebMessageOutputs.FindByParameters(temp).DataInfluencedBy(Find_Personal_Info());
+result = WebMessageOutputs.FindByParameters(temp).DataInfluencedBy(Find_Personal_Info()).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Medium_Threat / Client_HTML5_Insecure_Storage

Code changes

```
---
+++
@@ -9,7 +9,7 @@

 CxList potentialStoreData = storeData.FindByParameters(potential);
 potentialStoreData.Add(potential.FindByType<MemberAccess>());
-CxList storeInfluencedByInput = potentialStoreData.DataInfluencedBy(inputs);
+CxList storeInfluencedByInput = potentialStoreData.DataInfluencedBy(inputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 CxList sanitizer = Find_Encrypt();

 result.Add(storeInfluencedByInput.SanitizeCxList(sanitizer));
```

JavaScript / JavaScript_Medium_Threat / Client_Potential_Code_Injection

Code changes

```
---
+++
@@ -10,3 +10,5 @@
 result.Add(eval.FindByType<Param>() * inputs - sanitize,
         eval.InfluencedByAndNotSanitized(inputs, sanitize, CxList.InfluenceAlgorithmCalculation.NewAlgorithm),
         Find_Source_Equals_Sink(inputs, eval));
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Medium_Threat / Client_Potential_XSS

Code changes

```
---
+++
@@ -17,6 +17,7 @@
 CxList toParms = outputsParams
     .InfluencedByAndNotSanitized(inputs, sanitize, CxList.InfluenceAlgorithmCalculation.NewAlgorithm);

+CxList resultsList = All.NewCxList();
 foreach (CxList curPath in toParms.GetCxListByPath())
 {
     CxList lastNode = curPath.GetLastNodesInPath();
@@ -25,16 +26,55 @@
         CxList paramMethod = outMethods.FindByParameters(lastNode);
         if (paramMethod.Count == 1)
         {
-            result.Add(curPath.ConcatenatePath(paramMethod, false));
+            resultsList.Add(curPath.ConcatenatePath(paramMethod, false));
         }
     }
     else
     {
-        result.Add(curPath);
+        resultsList.Add(curPath);
     }
 }

-result.Add(outParams * inputs - sanitize,
-       Find_Source_Equals_Sink(inputs, outputs) - sanitize);
+resultsList.Add(outParams * inputs - sanitize,
+   Find_Source_Equals_Sink(inputs, outputs) - sanitize);

-result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
+//Lightning Component API since version 52 enforces Lightning locker by default which prevents XSS.
+//This will be considered as sanitizer if have metafile above version 52 with the same name as js file
+double safeVersion = 52.0;
+string xpathQuery =
+   "//*[local-name() = 'apiVersion' and (namespace-uri()='http://soap.sforce.com/2006/04/metadata' or namespace-uri()='')]";
+List<String> metaFiles = new List<string>();
+foreach (CxXmlDoc doc in cxXPath.GetXmlFiles("*.js-meta.xml"))
+{
+   XPathNavigator navigator = doc.CreateNavigator();
+   XPathNodeIterator nodeIterator = navigator.Select(xpathQuery);
+
+   while (nodeIterator.MoveNext())
+   {
+       string apiVersion = nodeIterator.Current.ToString();
+       double apiVersionValue = 0.0;
```

```
+
+       if (double.TryParse(apiVersion, out apiVersionValue))
+       {
+           if(apiVersionValue >= safeVersion)
+           {
+               metaFiles.Add(doc.FileName);
+           }
+       }
+   }
+}
+
+//remove results detected in sanitized file
+foreach (CxList res in resultsList.GetCxListByPath())
+{
+   CxList firstNode = res.GetFirstNodesInPath();
+   CSharpGraph g = firstNode.TryGetCSharpGraph<CSharpGraph>();
+   string fileName = g.LinePragma.FileName.Replace(".js", ".js-meta.xml");
+
+   if(metaFiles.Contains(fileName))
+   {
+       resultsList -= res;
+   }
+}
+
+result = resultsList.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Medium_Threat / Client_ReDoS_From_Regex_Injection

Code changes

```
---
+++
@@ -41,3 +41,5 @@
 result.Add(
     regex * evilStrings,
     regex.DataInfluencedBy(evilStrings));
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Medium_Threat / Client_ReDoS_In_Replace

Code changes

```
---
+++
@@ -23,3 +23,5 @@
 // Find relevant matches
 result = replace.InfluencedByAndNotSanitized(activeEvilRegexes, sanitize);
 result -= result.DataInfluencedBy(result);
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Medium_Threat / Client_Untrusted_Activex

Code changes

```
---

+++

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

 activex.Add(methods.FindByMemberAccess("Silverlight.createObject"));

 CxList inputs = Find_Inputs();


-result = activex.DataInfluencedBy(inputs);

+result = activex.DataInfluencedBy(inputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Medium_Threat / Client_XPATH_Injection

Code changes

```
---

+++

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

 CxList inputs = Find_Inputs();

 CxList evaluateXPATH = Find_Members("document.evaluate");

 CxList sanitizer = XPATH_Injection_Sanitize();

-result = evaluateXPATH.InfluencedByAndNotSanitized(inputs, sanitizer);

+result = evaluateXPATH.InfluencedByAndNotSanitized(inputs, sanitizer).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Medium_Threat / Frameable_Login_Page

Code changes

```
---

+++

@@ -110,3 +110,5 @@


 result = riskySendResponses.InfluencedByAndNotSanitized(riskyResponseParams, sanitizers);

 result.Add(riskyLoginHapiResponses);

+

+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Medium_Threat / Unchecked_Input_For_Loop_Condition

Code changes

```
---

+++

@@ -111,4 +111,4 @@

     assigns = assigns.GetAncOfType<AssignExpr>();

     sanitizers.Add(assigns.InfluencingOnAndNotSanitized(unboundLoopConditions, assignExprs - assigns).GetLastNodesInPath());

 }

-result = inputs.InfluencingOnAndNotSanitized(unboundLoopConditions, sanitizers);

+result = inputs.InfluencingOnAndNotSanitized(unboundLoopConditions, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Medium_Threat / XML_External_Entities_XXE

Code changes

```
---

+++
```

```
@@ -12,4 +12,4 @@
 CxList unsafeNoentOptions = fields.FindByShortName("noent") * trueAbstractValued.GetAssignee();
 CxList unsafelibXmljsMethods = methods.FindByMemberAccess("*.parseXml").DataInfluencedBy(unsafeNoentOptions);
 unsafelibXmljsMethods = unsafelibXmljsMethods.GetLastNodesInPath();
-result = inputs.DataInfluencingOn(unsafelibXmljsMethods);
+result = inputs.DataInfluencingOn(unsafelibXmljsMethods).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_ReactNative / Clipboard_Information_Leakage

Code changes

```
---
+++
@@ -3,5 +3,5 @@
    CxList outputs = ReactNative_Find_Outputs();
    outputs = outputs.FindByMemberAccess("Clipboard.setString");

-   result = inputs.InfluencingOn(outputs);
+   result = inputs.InfluencingOn(outputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

JavaScript / JavaScript_ReactNative / Insufficient_Transport_Layer_Security

Code changes

```
---
+++
@@ -35,4 +35,6 @@
    result.Add(insecureURLs,
            insecureURI);
+
+   result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

JavaScript / JavaScript_Server_Side_Vulnerabilities / Cookie_Poisoning

Code changes

```
---
+++
@@ -2,5 +2,6 @@
 CxList cookiesSet = Hapi_Find_Cookie_Set();
 CxList sanitize = NodeJS_Find_Encrypt();

-result = inputs.InfluencingOnAndNotSanitized(cookiesSet, sanitize);
+result = inputs.InfluencingOnAndNotSanitized(cookiesSet, sanitize)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 result.Add(inputs * cookiesSet - sanitize);
```

JavaScript / JavaScript_Server_Side_Vulnerabilities / CSRF

Code changes

```
---
+++
```

```
@@ -58,5 +58,5 @@
    updateDB = updateDB.FindByShortNames(new List<string>(findMethodsNames));
    CxList sanitizers = NodeJS_Find_General_Sanitize();
    CxList inputs = NodeJS_Find_Interactive_Inputs();
-   result = updateDB.InfluencedByAndNotSanitized(inputs, sanitizers);
+   result = updateDB.InfluencedByAndNotSanitized(inputs, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

JavaScript / JavaScript_Server_Side_Vulnerabilities / Dynamic_File_Inclusion

Code changes

```
---
+++
@@ -20,4 +20,4 @@

 CxList potentialParams = urMA.GetByAncs(requireLibrary);
 potentialParams.Add(createRequireParams);
-result = potentialParams.InfluencedByAndNotSanitized(inputs, sanitize);
+result = potentialParams.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Server_Side_Vulnerabilities / Information_Exposure_Through_an_Error_Message

Code changes

```
---
+++
@@ -56,5 +56,5 @@
 CxList hapiHandlers = Hapi_Find_Handler_Method_Under_Route(hapiRoutes);
 outputs.Add(hapiHandlers);

-result = outputs.InfluencedBy(errors);
+result = outputs.InfluencedBy(errors).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 result.Add(throwStatementsReturningToUser);
```

JavaScript / JavaScript_Server_Side_Vulnerabilities / JWT_Lack_Of_Expiration_Time

Code changes

```
---
+++
@@ -1,16 +1,14 @@
 // This query tries to find JWT tokens creation
 // where the expiration time is not defined at all
-
 CxList fields = Find_FieldDecls();
 CxList assocArrays = Find_AssociativeArrayExpr();
-
 CxList signMethods = Find_JsonWebToken_Sign();

 // Add calls without third parameter, as those also dont have expiration defined
 result = signMethods.FilterByDomProperty<MethodInvokeExpr>(x => x.Parameters.Count == 2);

-CxList expiresInKey = fields.FindByShortName("expiresIn").GetAssigner().GetByAncs(assocArrays);
```

```
+CxList expiresInKey = fields.FindByShortNames(new string[]{"expiresIn","exp"}).GetAssigner().GetByAncs(assocArrays);
 CxList sanitized = signMethods.DataInfluencedBy(expiresInKey)
-       .GetLastNodesInPath();
+   .GetLastNodesInPath();

 result.Add(signMethods - sanitized);
```

JavaScript / JavaScript_Server_Side_Vulnerabilities / MongoDB_NoSQL_Injection

Code changes

```
---
+++
@@ -54,5 +54,5 @@
 CxList replaceMethods = methods.FindByShortNames(new List<string> {"replace", "replaceAll"});
 sanitize.Add(replaceMethods.FindByParameters(sanRegex));

-result = outputs.InfluencedByAndNotSanitized(inputs, sanitize);
+result = outputs.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 result.Add(inputs * outputs - sanitize);
```

JavaScript / JavaScript_Server_Side_Vulnerabilities / Open_Redirect

Code changes

```
---
+++
@@ -2,5 +2,5 @@
 CxList inputs = NodeJS_Find_Interactive_Inputs();
 CxList sanitize = NodeJS_Find_Redirect_Sanitize();

-result = outputs.InfluencedByAndNotSanitized(inputs, sanitize);
+result = outputs.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 result.Add(inputs * outputs - sanitize);
```

JavaScript / JavaScript_Server_Side_Vulnerabilities / Parameter_Tampering

Code changes

```
---
+++
@@ -43,3 +43,5 @@
 CxList dbInput = db.DataInfluencedBy(input);
 result = methodWithoutAnd.DataInfluencedBy(dbInput, CxList.InfluenceAlgorithmCalculation.NewAlgorithm);
 result.Add(whereMeth);
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Server_Side_Vulnerabilities / Stored_Code_Injection

Code changes

```
---
+++
@@ -3,5 +3,6 @@
 CxList output = NodeJS_Find_Outputs_CodeInjection();
```

```
 CxList sanitize = NodeJS_Find_General_Sanitize();


-result = output.InfluencedByAndNotSanitized(inputs, sanitize, CxList.InfluenceAlgorithmCalculation.NewAlgorithm);
+result = output.InfluencedByAndNotSanitized(inputs, sanitize, CxList.InfluenceAlgorithmCalculation.NewAlgorithm)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 result.Add(inputs * output - sanitize);
```

JavaScript / JavaScript_Server_Side_Vulnerabilities / Stored_XSS

Code changes

```
---
+++
@@ -1,4 +1,5 @@
-CxList methods = Find_Methods();
+CxList methodInvokes = Find_Methods();
+CxList paramValue = Find_Param().CxSelectDomProperty<Param>(p => p.Value);


 //INPUTS (outputs from DB)
 CxList outputDB = All.NewCxList(
@@ -9,12 +10,12 @@
             NodeJS_Find_Read(),
             Find_Cloud_Storage_Inputs());


-CxList stats = All.FindByShortName("stat").FindByType<MethodInvokeExpr>();
-outputDB.Add(All.GetParameters(All.GetParameters(stats, 2), 1));
+CxList stats = methodInvokes.FindByShortName("stat");
+outputDB.Add(All.GetParameters(paramValue.GetParameters(stats, 2), 1));


 // SANITIZERS
 CxList sanitizers = All.NewCxList();
-sanitizers.Add(methods.FindByShortNames(new List<string> {"xss", "escape*","filter*","encode*"}));
+sanitizers.Add(methodInvokes.FindByShortNames(new []{"xss", "escape*","filter*","encode*"}));


 // SINKS
 CxList outputs = NodeJS_Find_Interactive_Outputs();
```

JavaScript / JavaScript_Server_Side_Vulnerabilities / Unrestricted_File_Upload

Code changes

```
---
+++
@@ -17,3 +17,5 @@
     Multer_FileUpload(),     //Multer uploads
     Loopback_FileUpload(),   //Loopback uploads
     Formidable_FileUpload());//Formidable uploads
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Server_Side_Vulnerabilities / Unsafe_Object_Binding

Code changes

```
---
+++
@@ -52,3 +52,5 @@
             inputObjects.GetParameters(entryCreatedFromInputObject).DataInfluencingOn(saveMethods),
             dbIn * inputObjects,
             dbIn.FindByParameters(inputObjects));
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Server_Side_Vulnerabilities / Use_of_Broken_or_Risky_Cryptographic_Algorithm

Code changes

```
---
+++
@@ -35,3 +35,5 @@
     Find_Unsafe_Encrypt(),
     cryptoMethods.DataInfluencedBy(hashStrings),
     createCipherMembers.DataInfluencedBy(badCiphers));
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

JavaScript / JavaScript_Vue / Vue_DOM_XSS

Code changes

```
---
+++
@@ -17,4 +17,6 @@
     CxList cxOnceIfs = Find_UnknownReference().FindByShortName("CxOnce").GetAncOfType<IfStmt>();
     CxList onceOutputs = outputs.GetByAncs(cxOnceIfs);
     result -= onceOutputs.InfluencedByAndNotSanitized(viewInputs, sanitize);
+
+    result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

JavaScript / JavaScript_XS / XS_Response_Splitting

Code changes

```
---
+++
@@ -29,5 +29,5 @@
             onLeftMail.FindByMemberAccess("Mail.*"));

     //look for all flows to vulnerable fields from inputs
-    result.Add(Fields.DataInfluencedBy(XS_Find_Interactive_Inputs()));
+    result.Add(Fields.DataInfluencedBy(XS_Find_Interactive_Inputs()).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow));
 }
```

JavaScript / JavasScript_Visualforce_Remoting / VF_Remoting_Client_Potential_Code_Injection

Code changes

```
---
+++
```

```
@@ -7,5 +7,6 @@
 paramsInputs -= sanitize;


 result.Add(paramsInputs,
-       (Eval).InfluencedByAndNotSanitized(inputs, sanitize, CxList.InfluenceAlgorithmCalculation.NewAlgorithm),
+       (Eval).InfluencedByAndNotSanitized(inputs, sanitize, CxList.InfluenceAlgorithmCalculation.NewAlgorithm)
+           .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow),
        Find_Source_Equals_Sink(inputs, Eval));
```

Lua / Lua_Medium_Threat / DoS_by_Sleep

Code changes

```
---
+++
@@ -1,11 +1,12 @@
 CxList methods = Find_Methods();
 CxList integers = Find_IntegerLiterals();
-CxList intsAndUnkRefs = All.NewCxList(integers, Find_UnknownReference());
+CxList realLiterals = Find_RealLiterals();
+CxList intsRealsAndUnkRefs = All.NewCxList(integers, Find_UnknownReference(), realLiterals);
 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);
+CxList intAndRealAbsValue = All.NewCxList(integers, realLiterals);
+CxList intAbsValue = intsRealsAndUnkRefs.FindByAbstractValues(intAndRealAbsValue);


 conditions = conditions.InfluencedBy(intAbsValue);
```

Objc / ObjectiveC_High_Risk / Information_Exposure_Through_Extension

Code changes

```
---
+++
@@ -27,4 +27,5 @@
 CxList targetMethods = methods.FindByShortNames(methodNames);


 CxList output = unkRefs.GetByAncs(parameters.GetParameters(targetMethods, 0));
-result = personalInfo.InfluencingOnAndNotSanitized(output, sanitize);
+result = personalInfo.InfluencingOnAndNotSanitized(output, sanitize)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Objc / ObjectiveC_High_Risk / Reflected_XSS_All_Clients

Code changes

```
---
+++
@@ -4,4 +4,5 @@
     outputs -= outputs.InfluencedBy(sourceApplicationCond);
```

```
 CxList sanitized = Find_XSS_Sanitize();


-result = outputs.InfluencedByAndNotSanitized(inputs, sanitized);
+result = outputs.InfluencedByAndNotSanitized(inputs, sanitized)
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Objc / ObjectiveC_High_Risk / Second_Order_SQL_Injection

Code changes

```
---
+++
@@ -3,7 +3,7 @@
 CxList sanitize = Find_SQL_Sanitize();
 CxList reads = Find_Read();

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

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

Objc / ObjectiveC_High_Risk / Stored_XSS

Code changes

```
---
+++
@@ -5,7 +5,7 @@
 CxList sanitize = Find_XSS_Sanitize();
 CxList reads = Find_Read();

-CxList dbRead = All.NewCxList();
-dbRead.Add(db, reads);
+CxList dbRead = All.NewCxList(db, reads);

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

Objc / ObjectiveC_High_Risk / Unsafe_Reflection

Code changes

```
---
+++
@@ -1,11 +1,12 @@
 CxList inputs = Find_Interactive_Inputs();
 CxList sanitized = Find_General_Sanitize();
 CxList methods = Find_Methods();
-string[] importNames = {
+string[] importNames = new [] {
```

```
    "NSClassFromString", "NSClassFromString:" /* Import class */,
    "NSSelectorFromString", "NSSelectorFromString:" /* Import selector */,
    "NSProtocolFromString", "NSProtocolFromString:" /* Import protocol */
    };
-CxList potentialReflection = methods.FindByShortNames(new List<string>(importNames));
+CxList potentialReflection = methods.FindByShortNames(importNames);

-result = potentialReflection.InfluencedByAndNotSanitized(inputs, sanitized);
+result = potentialReflection.InfluencedByAndNotSanitized(inputs, sanitized)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Objc / ObjectiveC_Low_Visibility / Allowed_Backup

Code changes

```
---
+++
@@ -24,7 +24,7 @@

 //Get the first parameters of CreationFile methods
 CxList createFileAtPathMethodsParams = All.GetParameters(createFileAtPathMethods, 0);
-createFileAtPathMethodsParams -= createFileAtPathMethodsParams.FindByType(typeof(Param));
+createFileAtPathMethodsParams -= createFileAtPathMethodsParams.FindByType<Param>();

 //Find Items without any configuration for backup
 foreach (CxList source in createFileAtPathMethodsParams){
@@ -34,7 +34,8 @@
 }

 //Find Items with ExcludedFromBackup set to No
-CxList setResourceValueFlows = PathsUrl.DataInfluencingOn(setResourceValueMethods);
+CxList setResourceValueFlows = PathsUrl.DataInfluencingOn(setResourceValueMethods)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

 foreach (CxList setResourceValueFlow in setResourceValueFlows.GetCxListByPath()){
     //Get the first node of the flow
@@ -43,7 +44,8 @@
     //Check if any creationFile method is influencing the setResourceValueFlow
     if(creation.Count > 0){
         //Concatenate the creationFile method with the setResourceValueFlow
-        result.Add(creation.ConcatenatePath(methods.FindByParameters(creation), false).ConcatenatePath(setResourceValueFlow, false));
+        result.Add(creation.ConcatenatePath(methods.FindByParameters(creation), false)
+            .ConcatenatePath(setResourceValueFlow, false));
     }
     else {
         //Otherwise add the setResourceValueFlow as a result
```

Objc / ObjectiveC_Low_Visibility / Information_Exposure_Through_an_Error_Message

Code changes

```
---
+++
@@ -1,4 +1,4 @@
 CxList catchTypes = Find_Catch();
 CxList ctch = All.FindAllReferences(catchTypes);
 CxList outputs = Find_Outputs();
-result = outputs.DataInfluencedBy(ctch);
+result = outputs.DataInfluencedBy(ctch).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Objc / ObjectiveC_Low_Visibility / Information_Leak_Through_Response_Caching

Code changes

```
---
+++
@@ -28,5 +28,6 @@
         .GetAncOfType(typeof(MethodInvokeExpr)).GetTargetOfMembers();
 
 
-    result = requests.InfluencingOnAndNotSanitized(responses, sanitizers);
+    result = requests.InfluencingOnAndNotSanitized(responses, sanitizers)
+        .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

Objc / ObjectiveC_Low_Visibility / Log_Forging

Code changes

```
---
+++
@@ -2,4 +2,5 @@
 CxList logs = Find_Log_Outputs();
 CxList sanitize = Find_General_Sanitize();
 
-result = logs.InfluencedByAndNotSanitized(inputs, sanitize);
+result = logs.InfluencedByAndNotSanitized(inputs, sanitize)
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Objc / ObjectiveC_Low_Visibility / Plain_Text_Transport_Layer

Code changes

```
---
+++
@@ -1,3 +1,3 @@
 CxList pureHttp = Find_Pure_Http();
 CxList newOutputs  =  Find_Insufficient_Out_Transport_Layer();
-result = newOutputs.InfluencedBy(pureHttp);
+result = newOutputs.InfluencedBy(pureHttp).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Objc / ObjectiveC_Low_Visibility / Poor_Authorization_and_Authentication

Code changes

```
---
+++
```

```
@@ -12,4 +12,5 @@
 CxList identifiers = All.NewCxList();
 identifiers.Add(UDID, IP, All.FindByShortName("*MacAddress*", false));

-result = identifiers.InfluencingOnAndNotSanitized(outputs, Find_General_Sanitize());
+result = identifiers.InfluencingOnAndNotSanitized(outputs, Find_General_Sanitize())
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Objc / ObjectiveC_Low_Visibility / Use_of_Hardcoded_Cryptographic_Key

Code changes

```
---
+++
@@ -2,5 +2,6 @@
 CxList strings = Find_Strings();
 CxList bin = Find_BinaryExpr();

-result = param3.FindByType(typeof(StringLiteral));
-result.Add(param3.InfluencedByAndNotSanitized(strings, bin));
+result.Add(
+    param3.FindByType<StringLiteral>(),
+    param3.InfluencedByAndNotSanitized(strings, bin).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow));
```

Objc / ObjectiveC_Medium_Threat / Cut_And_Paste_Leakage

Code changes

```
---
+++
@@ -14,7 +14,7 @@
 // Find the functions that run right before entering background.
 CxList enterBackground = Find_EnterBackground_Statements();

-List<string> methodsNames = new List<string>{"setString", "string"};
+string[] methodsNames = new string[]{"setString", "string"};
 CxList setString = All.FindByShortNames(methodsNames);

 //     [[UIPasteboard generalPasteboard] setString:@""]
@@ -96,7 +96,7 @@

     CxList output = Find_Interactive_Outputs_User();
     output -= output.FindByType(@"UILabel");
-    result.Add(personalInfo.DataInfluencingOn(output));
+    result.Add(personalInfo.DataInfluencingOn(output).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow));

     definitions = definitions.FindDefinition(personalInfo - output);
     result.Add(definitions);
```

Objc / ObjectiveC_Medium_Threat / Insufficient_Transport_Layer_Input

Code changes

```
---
+++
@@ -6,4 +6,5 @@
 CxList personalInfo = Find_Personal_Info();
 CxList unsecureInputs = Find_Pure_Http_and_Downloaded_Data();

-result = personalInfo.DataInfluencedBy(unsecureInputs, CxList.InfluenceAlgorithmCalculation.NewAlgorithm);
+result = personalInfo.DataInfluencedBy(unsecureInputs, CxList.InfluenceAlgorithmCalculation.NewAlgorithm)
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

PHP / PHP_High_Risk / Reflected_XSS

Code changes

```
---
+++
@@ -1,18 +1,9 @@
-CxList methods = Find_Methods();
-
 CxList inputs = Find_Interactive_Inputs();
 CxList outputs = Find_Interactive_Outputs();

-// `readfile` automatically outputs the contents: it's not a sanitizer
-CxList readfile = methods.FindByShortName("readfile", false);
-readfile.Add(
-    Find_Params().CxSelectDomProperty<Param>(p => p.Value).GetParameters(readfile, 0)
-);
-
 CxList sanitized = All.NewCxList(
     Find_XSS_Sanitize(),
-    Find_File_Access() - readfile
-);
+    Find_File_Access());

 if(All.useConsoleInputs)
 {
```

PHP / PHP_High_Risk / Stored_XSS

Code changes

```
---
+++
@@ -1,15 +1,17 @@
 CxList outputs = Find_Interactive_Outputs();
 CxList sanitizers = Find_XSS_Sanitize();
-CxList stored = All.NewCxList();
-stored.Add(
+CxList stored = All.NewCxList(
     Find_DB_Out(),
-    Find_Stored_Inputs()
-);
```

```
+   Find_Stored_Inputs());


 result.Add(
    // Duplicated nodes
    (stored * outputs) - sanitizers,
    // Flow
-   stored.InfluencingOnAndNotSanitized(outputs, sanitizers)
-       .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow)
-);
+   outputs.InfluencedByAndNotSanitized(stored, sanitizers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow)
+   );
+
+//Removing invalid flow through _getBMPimage method (Only 2nd param is valid)
+CxList invalidFlow = Find_Param().CxSelectDomProperty<Param>(p => p.Value).GetParameters(
+   Find_Methods().FindByShortNames(new [] {"_getBMPimage", "imagebmp"}, false), 0);
+result -= result.IntersectWithNodes(invalidFlow);
```

PHP / PHP_Medium_Threat / Hashing_Length_Extension_Attack

Code changes

```
---
+++
@@ -44,6 +44,9 @@
 // Sinks
 // - shorthand methods
 CxList sinks = methods.FindByShortNames(new[]{"sha1", "md5"}, false);
+// - Filter by sinks that have a concat (with a secret) in it's parameters
+sinks = concats.GetParameters(sinks).GetAncOfType<MethodInvokeExpr>();
+
 // - hash calls
 CxList hash = methods.FindByShortName("hash", false);
 CxList hashCalls = algoStrings.GetParameters(hash, 0).GetAncOfType<MethodInvokeExpr>();
@@ -67,5 +70,5 @@
 result.Add(
    unsafeWithPreviousSafeUpdate.NotInfluencingOn(safeHashUpdates).GetLastNodesInPath().GetAncOfType<MethodInvokeExpr>(),
    inputs.InfluencingOnAndNotSanitized(sinks, sanitizers).IntersectWithNodes(userRight)
-       .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow)
-);
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow)
+   );
```

PHP / PHP_Medium_Threat / Header_Injection

Code changes

```
---
+++
@@ -4,6 +4,10 @@
 CxList sanitize = Find_Header_Injection_Sanitize();
 sanitize.Add(Find_General_Sanitize());
 CxList unkRefs = Find_UnknownReferences();
```

```
+

+CxList curlOptHeaderUnkRefs = unkRefs.FindByShortName("CURLOPT_HTTPHEADER");

+CxList safeCurlSetOptMethods = methods.FindByShortName("curl_setopt", false);

+safeCurlSetOptMethods -= safeCurlSetOptMethods.FindByParameters(curlOptHeaderUnkRefs);


 // Create lists to be used in query

 CxList taintedParamRefs = All.NewCxList();

@@ -23,6 +27,7 @@

 filterParameters = (methodNames, paramIndex) =>

 {

    CxList headerMethods = methods.FindByShortNames(methodNames, false);

+   headerMethods -= safeCurlSetOptMethods;

    return unkRefs.GetParameters(headerMethods, paramIndex);

 };
```

PHP / PHP_Medium_Threat / Open_Redirect

Code changes

```
---

+++

@@ -41,5 +41,8 @@

 // Add allowed hosts and header influenced by str_starts_with/in_array to the sanitize list

 sanitize.Add(Find_UnknownReference().GetSanitizerByMethodInCondition(sanitize).GetAncOfType<MethodInvokeExpr>());


+// Add redirects to relative URLs to the sanitize list

+sanitize.Add(redirectLocationStrings.FilterByDomProperty<StringLiteral>(x => x.Value.EndsWith("=")).GetAncOfType<MethodInvokeExpr>());

+

 // Get all redirect functions influenced by inputs/sanitizers  and add to result

 result = redirectFunctions.InfluencedByAndNotSanitized(Find_Interactive_Inputs(), sanitize);
```

PHP / PHP_Medium_Threat / Value_Shadowing

Code changes

```
---

+++

@@ -11,6 +11,7 @@

 {

    // filter lists

    CxList indexerUsages = indexerRef1.GetByAncs(trueStmt);

+   indexerUsages -= indexerUsages.GetByAncs(ifCondition);

    CxList indexerInCondition = indexerRef2.GetByAncs(ifCondition);


    CxList valueShadowing = All.NewCxList();
```

Python / Python_AWS_Lambda / AWS_Credentials_Leak

Code changes

```
---

+++

@@ -16,5 +16,5 @@
```

```
    inputs.Add(envirVars.FilterByDomProperty<IndexerRef>(x => x.Indices[0] is StringLiteral index &&
        envirVarsKeys.Contains(index.Value)));

-   result.Add(inputs.InfluencingOn(outputs));
+   result.Add(inputs.InfluencingOn(outputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow));
 }
```

Python / Python_AWS_Lambda / DynamoDB_NoSQL_Injection

Code changes

```
---
+++
@@ -43,5 +43,5 @@

    CxList paramsInfluenced = All.NewCxList();
    paramsInfluenced.Add(statementParam, statementParamArray, filterExprParam);
-   result = paramsInfluenced.InfluencedBy(inputs);
+   result = paramsInfluenced.InfluencedBy(inputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

Python / Python_AWS_Lambda / Hardcoded_AWS_Credentials

Code changes

```
---
+++
@@ -21,7 +21,7 @@
        .CxSelectDomProperty<Param>(_ => _.Value);

    // Flows from string literals to parameters
-   result = keyParams.InfluencedBy(strings);
+   result = keyParams.InfluencedBy(strings).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

    // Hardcoded strings in the parameter
    result.Add(keyParams * strings);
```

Python / Python_AWS_Lambda / Use_of_Hardcoded_Cryptographic_Key_On_Server

Code changes

```
---
+++
@@ -106,7 +106,7 @@
        .CxSelectDomProperty<Param>(_ => _.Value);

    // Flows from string literals to parameters
-   result = keyParams.InfluencedBy(strings);
+   result = keyParams.InfluencedBy(strings).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

    // Hardcoded strings in the parameter
    result.Add(keyParams * strings);
```

Python / Python_High_Risk / Command_Injection

Code changes

```
---
+++
@@ -26,7 +26,8 @@
 CxList commands = Find_Command_Execution();
 String[] osMethods = new string[] {"input"};
 CxList osMethodsList = Find_Methods_By_Import("os", osMethods, imports);
-result = commands.InfluencedByAndNotSanitized(inputs, sanitize);
+result = commands.InfluencedByAndNotSanitized(inputs, sanitize)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

 CxList initialValuesValidInputs = result.IntersectWithNodes(validNodes);
 result -= result.IntersectWithNodes(invalidNodes);
```

Python / Python_High_Risk / LDAP_Injection

Code changes

```
---
+++
@@ -3,4 +3,5 @@
 CxList outputs = Find_LDAP_Inputs();
 CxList sanitize = Find_LDAP_Sanitize();

-result = outputs.InfluencedByAndNotSanitized(inputs, sanitize);
+result = outputs.InfluencedByAndNotSanitized(inputs, sanitize)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Python / Python_High_Risk / OS_Access_Violation

Code changes

```
---
+++
@@ -27,4 +27,5 @@

 CxList insecureMethodsParameters = unkrefs.GetParameters(insecureMethods);

-result = insecureMethodsParameters.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = insecureMethodsParameters.InfluencedByAndNotSanitized(inputs, sanitizers)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Python / Python_High_Risk / Second_Order_SQL_Injection

Code changes

```
---
+++
@@ -21,4 +21,5 @@
    pyodbcRefs,
    execute);

-result = dbOutRead.InfluencingOnAndNotSanitized(dbIn, sanitize);
```

```
+result = dbOutRead.InfluencingOnAndNotSanitized(dbIn, sanitize)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Python / Python_High_Risk / Unsafe_Deserialization

Code changes

```
---
+++
@@ -1,26 +1,25 @@
 CxList methods = Find_Methods();

-CxList insecureMethods = Find_Methods_By_Import("pandas", new string[]{"read_pickle"});
+CxList insecureMethods = All.NewCxList(
+   Find_Methods_By_Import("pandas", new string[]{"read_pickle"}),
+   Find_Methods_By_Import("yaml", new string[]{"load"}),
+   methods.FindByMemberAccesses("pickle", new string[] {"load", "loads", "noload", "Unpickler"}),
+   methods.FindByMemberAccess("shelve.open"));

-List<string> pickleMethodNames = new List<string>{"load", "loads", "noload", "Unpickler"};
-CxList pickleMemberAccess = methods.FindByMemberAccess("pickle.*");
-insecureMethods.Add(pickleMemberAccess.FindByShortNames(pickleMethodNames));
-insecureMethods.Add(methods.FindByMemberAccess("shelve.open"));
-
-List<string> hashMethodNames = new List<string>{"md5", "sha256"};
-CxList hashMemberAccess = methods.FindByMemberAccess("hashlib.*");
-CxList hash = hashMemberAccess.FindByShortNames(hashMethodNames);
+CxList hash = methods.FindByMemberAccesses("hashlib", new string[] {"md5", "sha256"});

 CxList IfsWithInsecureMethods = insecureMethods.GetAncOfType<IfStmt>();
 CxList conditions = All.FindByFathers(IfsWithInsecureMethods).FindByType<Expression>();
 CxList safeConditions = conditions.DataInfluencedBy(hash).GetLastNodesInPath();
 insecureMethods -= insecureMethods.GetByAncs(safeConditions.GetAncOfType<IfStmt>());

-CxList inputs = All.NewCxList(Find_Inputs() - Find_Invalid_Django_Injection_Inputs(),
-   Find_Read(), Find_Cloud_Inputs());
+CxList inputs = All.NewCxList(
+   Find_Inputs() - Find_Invalid_Django_Injection_Inputs(),
+   Find_Read(),
+   Find_Cloud_Inputs());
 inputs -= insecureMethods;

-CxList sanitizers = Find_Sanitize();
-sanitizers.Add(Find_Base64_Encode());
+CxList sanitizers = All.NewCxList(Find_Sanitize(), Find_Base64_Encode());

-result = insecureMethods.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = insecureMethods.InfluencedByAndNotSanitized(inputs, sanitizers)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Python / Python_Low_Visibility / Log_Forging

Code changes

```
---
+++
@@ -3,4 +3,5 @@

 CxList sanitize = Find_Integers();

-result = logs.InfluencedByAndNotSanitized(inputs, sanitize);
+result = logs.InfluencedByAndNotSanitized(inputs, sanitize)
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Python / Python_Low_Visibility / Marshmallow_Dumping_Without_Validation

Code changes

```
---
+++
@@ -17,4 +17,5 @@
 CxList sanitizers = methods.FindByShortNames("validate", "load");
 relevantSinks = relevantSinks - relevantSinks.InfluencingOn(sanitizers).GetFirstNodesInPath();

-result = inputs.InfluencingOnAndNotSanitized(relevantSinks, sanitizers);
+result = inputs.InfluencingOnAndNotSanitized(relevantSinks, sanitizers)
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Python / Python_Low_Visibility / Trust_Boundary_Violation_in_Session_Variables

Code changes

```
---
+++
@@ -4,4 +4,6 @@
 CxList sanitizers = Find_DB_In();
 sanitizers.Add(Find_Read(), Find_Django_QuerySet_Methods());

-result = session.InfluencedByAndNotSanitized(inputs, sanitizers, CxList.InfluenceAlgorithmCalculation.NewAlgorithm);
+result = session.InfluencedByAndNotSanitized(inputs, sanitizers,
+    CxList.InfluenceAlgorithmCalculation.NewAlgorithm)
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Python / Python_Medium_Threat / DoS_by_Sleep

Code changes

```
---
+++
@@ -6,4 +6,5 @@
 sleep.Add(Find_Members("time.sleep", methods));
 sleep.Add(Find_Methods_By_Import("time", new string[]{"sleep"}));

-result = sleep.DataInfluencedBy(inputs);
+result = sleep.DataInfluencedBy(inputs)
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Python / Python_Medium_Threat / Filtering_Sensitive_Logs

Code changes

```
---
+++
@@ -2,4 +2,5 @@
 CxList personalInfo = Find_Personal_Info();
 CxList sanitizers = Find_FilteringSensitiveLogsSanitizers();

-result = personalInfo.InfluencingOnAndNotSanitized(logs, sanitizers);
+result = personalInfo.InfluencingOnAndNotSanitized(logs, sanitizers)
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Python / Python_Medium_Threat / Header_Injection

Code changes

```
---
+++
@@ -22,4 +22,5 @@

 CxList inputs = All.NewCxList(Find_Inputs() - Find_Invalid_Django_Injection_Inputs(), Find_Cloud_Inputs());

-result = requests.DataInfluencedBy(inputs);
+result = requests.DataInfluencedBy(inputs)
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Python / Python_Medium_Threat / Improper_Restriction_of_XXE_Ref

Code changes

```
---
+++
@@ -3,7 +3,10 @@
 Find all the XML parser readers that are not
 sanitized for the XXE vulnerability.
 ************************/
-result = Find_XXE_Expat();
-result.Add(Find_XXE_Sax());
-result.Add(Find_XXE_SaxSanitized_XMLReaders());
-result.Add(Find_XXE_lxml());
+CxList improperXXE_Ref = All.NewCxList(
+    Find_XXE_Expat(),
+    Find_XXE_Sax(),
+    Find_XXE_SaxSanitized_XMLReaders(),
+    Find_XXE_lxml());
+
+result = improperXXE_Ref;
```

Python / Python_Medium_Threat / Object_Access_Violation

Code changes

```
---
+++
```

```
@@ -22,4 +22,5 @@
 CxList sanitizedMethods = attrMethods.FindByParameters(sanitizingParams);
 attrMethods -= sanitizedMethods;

-result = attrMethods.InfluencedByAndNotSanitized(inputs, sanitize);
+result = attrMethods.InfluencedByAndNotSanitized(inputs, sanitize)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

Python / Python_Medium_Threat / ReDoS_Injection

Code changes

```
---
+++
@@ -3,4 +3,5 @@
 sanitizers.Add(Find_Base64_Encode());
 CxList regexPaternParams = Find_Regex_Pattern_Params();

-result = regexPaternParams.InfluencedByAndNotSanitized(inputs, sanitizers);
+result = regexPaternParams.InfluencedByAndNotSanitized(inputs, sanitizers)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

RPG / RPG_Low_Visibility / Use_Of_Hardcoded_Password

Code changes

```
---
+++
@@ -6,14 +6,14 @@
 CxList pswInLSide = psw.FindByAssignmentSide(CxList.AssignmentSide.Left);
 CxList pswInLSideDecl = pswInLSide.FindByType<Declarator>();
 CxList strLiterals = Find_Strings();
+
+strLiterals -= Find_Empty_Strings();
+//when the hardcoded string includes a space or dot we believe
+//it is not a password string
+strLiterals -= strLiterals.FindByNames(new []{"* *","*.*","*/*","*\\*"});
+
 CxList litInRSide = strLiterals.FindByAssignmentSide(CxList.AssignmentSide.Right);
 litInRSide.Add(strLiterals.GetParameters(moves, 0));
-//when the hardcoded string includes a space or dot we believe
-//it is not a password string
-litInRSide -= litInRSide.FindByNames(new []{"* *","*.*","*/*","*\\*"});
-
-//empty string is OK
-litInRSide -= Find_Empty_Strings();

 // Password in declaration
 CxList PasswordInDecl = pswInLSideDecl.FindByInitialization(litInRSide);
@@ -33,11 +33,11 @@
 }
 PasswordInDecl -= notHdPass;
```

```
-
 // Find password in a '==' operator
 CxList EqualBinaryExpr = psw.GetFathers().FindByType<BinaryExpr>().
    GetByBinaryOperator(Checkmarx.Dom.BinaryOperator.IdentityEquality);
 CxList EqualOperatorStrings = All.NewCxList();
+
 foreach(CxList bin in EqualBinaryExpr)
 {
    CxList password = psw.FindByFathers(bin);
```

Rust / Rust_Low_Visibility / Privacy_Violation_in_Files

Code changes

```
---
+++
@@ -1,8 +1,6 @@
 CxList personalInfo = Find_Personal_Info();
-
 CxList sinks = Find_File_Write();
-
-CxList sanitizers = Find_Output_Sanitizers();
+CxList sanitizers = Find_Sensitive_Information_Sanitizers();

 result = sinks.InfluencedByAndNotSanitized(personalInfo, sanitizers)
    .ReduceFlow(CxList.ReduceFlowType.ReduceSmallFlow);
```

Rust / Rust_Low_Visibility / Privacy_Violation_in_Logs

Code changes

```
---
+++
@@ -1,21 +1,10 @@
-CxList methods = Find_Methods();
 CxList exprs = Find_Expressions();
 CxList personalInfo = Find_Personal_Info();
-CxList sanitizers = Find_Output_Sanitizers();
+CxList sanitizers = Find_Sensitive_Information_Sanitizers();
+sanitizers.Add(Find_Error_Variables());

-CxList logMethods = methods.FindByShortNames("info", "warn", "error", "debug", "trace", "log");
+CxList logMethods = Find_Log_Outputs();
 CxList sinks = exprs.GetParameters(logMethods);
-
-CxList errVars = Find_TypeTestExpr()
-   .FilterByDomProperty<TypeTestExpr>(_ => _.Type.ShortName == "Err")
-   .CxSelectDomProperty<TypeTestExpr>(_ => _.Expression);
-
-CxList errVarsIfs = errVars.GetFathers().GetFathers();
-
```

```
-CxList variableInitializers = errVarsIfs.CxSelectDomProperty<IfExpr>(_ => _.VariablesInitializers);
-variableInitializers.Add(errVarsIfs.CxSelectDomProperty<IfStmt>(_ => _.VariablesInitializers));
-
-sanitizers.Add(exprs.FindAllReferences(errVars).GetByAncs(variableInitializers));


 result = sinks.InfluencedByAndNotSanitized(personalInfo, sanitizers)
     .ReduceFlow(CxList.ReduceFlowType.ReduceSmallFlow);
```

Rust / Rust_Medium_Threat / Empty_Password_In_Connection_String

Code changes

```
---
+++
@@ -12,7 +12,7 @@
     CxList imports = members.FindByNames(importsNames);
     imports.Add(expressions.FindAllReferences(imports.GetAssignee()));
     imports.Add(vars.FindByNames(importsNames));
-    imports.Add(expressions.FindAllReferences(vars));
+    imports.Add(expressions.FindAllReferences(vars.FindByNames(importsNames)));
     return imports;
     };


@@ -24,7 +24,7 @@
     };

  //Mongo
-string[] mongoImports = new[]{"mongodb.sync.Client", "mongodb.Client"};
+string[] mongoImports = new[]{"mongodb.sync.Client", "mongodb.Client", "*Client"};
  string[] mongoMethods = new[]{"with_uri_str", "with_options"};
  CxList mongoSinks = GetMembers(mongoImports, mongoMethods);


@@ -40,11 +40,11 @@
  string[] sqlMethods = new[]{"connect", "connect_with"};

  CxList sqlSinks = GetMembers(sqlImports, sqlMethods);
-CxList mySqlConnOpts = GetImports(new[]{"sqlx.mysql.MySqlConnectOptions"});
-
+CxList mySqlConnOpts = GetImports(new[]{"*.MySqlConnectOptions", "*.MySqlConnection"});
  CxList password = methods.FindByShortName("password").GetMembersWithTargets(mySqlConnOpts, 7);
  CxList connect = methods.FindByShortName("connect").GetMembersWithTargets(mySqlConnOpts, 7);
  sqlSinks.Add(connect);
+

  CxList sinks = All.NewCxList(mongoSinks, dieselSinks, sqlSinks);
```

Rust / Rust_Medium_Threat / Encoding_Used_Instead_of_Encryption

Code changes

```
---
+++
@@ -1,4 +1,4 @@
-CxList sensitiveInfo = Find_Personal_Info();
+CxList sensitiveInfo = Find_Sensitive_Information();

 CxList encoders = Find_Encoders();
```

Rust / Rust_Medium_Threat / Privacy_Violation

Code changes

```
---
+++
@@ -1,15 +1,9 @@
 //Find_Personal_Info does not include e-mail
 CxList inputs = Find_Personal_Info();

-
 CxList outputs = Find_Interactive_Outputs();

-CxList sanitizers = All.NewCxList(
-   Find_Encrypt(),
-   Find_KDF(),
-   Find_Weak_or_Broken_KDF(),
-   Find_Hash(),
-   Find_Weak_or_Broken_Hash());
+CxList sanitizers = Find_Sensitive_Information_Sanitizers();

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

Rust / Rust_Medium_Threat / SSRF

Code changes

```
---
+++
@@ -15,34 +15,13 @@
     return exprs.GetParameters(invoke, pOrder);
     });

-
 CxList outputs = All.NewCxList(getSockets("TcpStream", "connect", 0));
-outputs.Add(getSockets("reqwest", "get", 0));
-outputs.Add(getSockets("UdpSocket", "connect", 0));
-outputs.Add(getSockets("blocking", "get", 0));
-outputs.Add(getSockets("Request", "new", 1));
-
-CxList reqwestClients = members.FindByNames("reqwest.Client", "reqwest.blocking.Client")
-   .GetMembersOfTarget().FindByShortName("new");
```

```
-outputs.Add(unkRefs.GetParameters(reqwestClients.GetMembersOfTarget().FindByShortName("get")));
-
-CxList reqwestRefs = unkRefs.FindAllReferences(reqwestClients.GetAncOfType<Declarator>())
-   .GetMembersOfTarget().FindByShortName("get");
-outputs.Add(unkRefs.GetParameters(reqwestRefs));
-
-CxList reqwestMembers = exprs.GetMembersWithTargets(unkRefs.FindByShortName("reqwest"), 5);
-outputs.Add(exprs.GetParameters(reqwestMembers.FindByShortNames(
-   "delete",
-   "get",
-   "head",
-   "patch",
-   "post",
-   "put"
-   ), 0));
-
-outputs.Add(exprs.GetParameters(reqwestMembers.FindByShortNames(
-   "request"
-   ), 1));
+outputs.Add(
+   getSockets("reqwest", "get", 0),
+   getSockets("UdpSocket", "connect", 0),
+   getSockets("blocking", "get", 0),
+   getSockets("Request", "new", 1),
+   Find_Reqwest_Outputs());

 CxList sanitizers = Find_WhiteListSanitizers();
```

VbNet / VbNet_Best_Coding_Practice / Dynamic_SQL_Queries

Code changes

```
---
+++
@@ -2,7 +2,7 @@
 CxList dbMethods = All.GetMethod(db); // All methods that contain a DB (others are usually irrelevant)
 CxList vbDB = All.GetByAncs(dbMethods);
 CxList methods = Find_Methods();
-CxList methDeclInvoke = All.FindByType(typeof(MethodDecl));
+CxList methDeclInvoke = Find_MethodDecls();
 methDeclInvoke.Add(methods);

 // Add all methods called by the DB methods - up to 3 levels of calls
@@ -13,13 +13,12 @@
    vbDB.Add(All.GetByAncs(dbDelta));
 }

-CxList binary = vbDB.FindByType(typeof(BinaryExpr));
+CxList binary = vbDB.FindByType<BinaryExpr>();
```

```
 CxList stringMethods = (vbDB * methods).FindAllReferences(All.FindByReturnType("String", false));
 CxList append = vbDB.FindByShortName("Append", false);

-CxList str = (vbDB.FindByType(typeof(UnknownReference)) +
-   vbDB.FindByType(typeof(Declarator))).FindByType("String", false);
+CxList str = All.NewCxList(vbDB.FindByType<UnknownReference>(), vbDB.FindByType<Declarator>()).FindByType("String", false);
 str.Add(stringMethods);

 CxList concat = binary.DataInfluencingOn(str);
@@ -30,4 +29,4 @@
 CxList substring = methods.FindByMemberAccess("String.Substring", false);
 sanitize.Add(All.GetByAncs(All.GetParameters(substring)));

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

VbNet / VbNet_Best_Coding_Practice / Hardcoded_Connection_String

Code changes

```
---
+++
@@ -1,6 +1,6 @@
 CxList conStr = Find_Strings().FindByName("*provider*", false);

-CxList openConnection = All.FindByType(typeof(ObjectCreateExpr)).FindByShortName("*connection", false);
+CxList openConnection = Find_ObjectCreations().FindByShortName("*connection", false);
 CxList openConParam = All.GetParameters(openConnection);

-result.Add(conStr.DataInfluencingOn(openConParam));
+result = conStr.DataInfluencingOn(openConParam).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Best_Coding_Practice / PersistSecurityInfo_is_True

Code changes

```
---
+++
@@ -1,6 +1,7 @@
 CxList PersistSecurityInfo = All.FindByRegex(@"persist security info(\s)*=(\s)*(true|yes)");

-CxList openConnection = All.FindByType(typeof(ObjectCreateExpr)).FindByShortName("*connection", false);
+CxList openConnection = Find_ObjectCreations().FindByShortName("*connection", false);
 CxList openConParam = All.GetParameters(openConnection);

-result = openConParam.DataInfluencedBy(PersistSecurityInfo.FindByType(typeof(StringLiteral)));
+result = openConParam.DataInfluencedBy(PersistSecurityInfo.FindByType<StringLiteral>())
+    .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);;
```

VbNet / VbNet_High_Risk / Code_Injection

Code changes

```
---
+++
@@ -1,9 +1,7 @@
-CxList CodeProvider = All.FindByMemberAccess("CSharpCodeProvider.*", false);
-CodeProvider.Add(All.FindByMemberAccess("VBCodeProvider.*", false));
-CodeProvider.Add(All.FindByMemberAccess("JScriptCodeProvider.*", false));
-CodeProvider.Add(All.FindByMemberAccess("CodeDomProvider.*", false));
-//find only codeCompilers (clean anrelevant methods)
-List < string > methCodeCompilers = new List<string> {"CompileAssemblyFrom*", "Parse"};
-CxList codeCompilers = CodeProvider.FindByShortNames(methCodeCompilers, false);
+//find only codeCompilers (clean and relevant methods)
+CxList codeCompilers = Find_Methods().FindByMemberAccesses(
+   new [] {"CSharpCodeProvider", "VBCodeProvider", "JScriptCodeProvider", "CodeDomProvider"},
+   new [] {"CompileAssemblyFrom*", "Parse"},
+   false);

-result = Find_Interactive_Inputs().DataInfluencingOn(codeCompilers);
+result = Find_Interactive_Inputs().DataInfluencingOn(codeCompilers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_High_Risk / Command_Injection

Code changes

```
---
+++
@@ -1,4 +1,5 @@
 CxList inputs = Find_Interactive_Inputs();
 CxList exec = Find_Command_Execution();
 CxList sanitize = Find_Command_Injection_Sanitize();
-result = exec.InfluencedByAndNotSanitized(inputs, sanitize);
+
+result = exec.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_High_Risk / Connection_String_Injection

Code changes

```
---
+++
@@ -3,4 +3,4 @@

 CxList sanitize = Find_General_Sanitize();

-result = con.InfluencedByAndNotSanitized(inputs, sanitize);
+result = con.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_High_Risk / Dangerous_File_Upload

Code changes

```
---
+++
@@ -3,5 +3,5 @@
 CxList save = All.FindByName("*.PostedFile.SaveAs", false);
```

```
 CxList MapPath = All.FindByName("Server.MapPath", false);


-result = save.DataInfluencedBy(file + inputs) -
-           save.DataInfluencedBy(MapPath);
+result = save.DataInfluencedBy(All.NewCxList(file, inputs)) - save.DataInfluencedBy(MapPath);
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_High_Risk / Reflected_XSS_All_Clients

Code changes

```
---
+++
@@ -10,5 +10,5 @@


    CxList sanitized = Find_XSS_Sanitize();


-   result = All.FindXSS(inputs, outputs, sanitized);
+   result = All.FindXSS(inputs, outputs, sanitized).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

VbNet / VbNet_High_Risk / Resource_Injection

Code changes

```
---
+++
@@ -1,2 +1,2 @@
-CxList socket = All.FindByType(typeof(ObjectCreateExpr)).FindByShortName("TcpListener", false);
-result = Find_Interactive_Inputs().DataInfluencingOn(All.GetByAncs(socket));
+CxList socket = Find_ObjectCreations().FindByShortName("TcpListener", false);
+result = Find_Interactive_Inputs().DataInfluencingOn(All.GetByAncs(socket)).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_High_Risk / Second_Order_SQL_Injection

Code changes

```
---
+++
@@ -3,5 +3,5 @@


 CxList dbParams = Find_DB_In();


-result = All.FindSQLInjections(db + Find_Read() - All.FindByShortName("ExecuteNonQuery", false),
-   dbParams, sanitize);
+result = All.FindSQLInjections(All.NewCxList(db, Find_Read()) - All.FindByShortName("ExecuteNonQuery", false),
+   dbParams, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_High_Risk / Stored_XSS

Code changes

```
---
+++
@@ -4,5 +4,5 @@
    CxList outputs = Find_XSS_Outputs();
```

```
    CxList sanitize =  Find_XSS_Sanitize();


-   result = All.FindXSS(db + Find_Read(), outputs, sanitize);
+   result = All.FindXSS(All.NewCxList(db, Find_Read()), outputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

VbNet / VbNet_High_Risk / XPath_Injection

Code changes

```
---
+++
@@ -3,4 +3,4 @@
 CxList inputs = Find_Interactive_Inputs();
 CxList sanitized = Find_Sanitize();

-result = XPath.InfluencedByAndNotSanitized(inputs, sanitized);
+result = XPath.InfluencedByAndNotSanitized(inputs, sanitized).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Low_Visibility / Impersonation_Issue

Code changes

```
---
+++
@@ -10,12 +10,12 @@
 //only parameters of logonuser methods that are influenced by inputs
 CxList LogonParams = All.GetParameters(Logon);     //all parameters of LogonUser method
 CxList LogonParamsIn = LogonParams.DataInfluencedBy(Inputs);   //parameters of LogonUser method that influenced by input
-LogonParamsIn = LogonParams.GetStartAndEndNodes(CxList.GetStartEndNodesType.EndNodesOnly);
+LogonParamsIn = LogonParams.GetLastNodesInPath();

 //only parameters of duplicatetoken methods that are influenced by logonuser methods parameters
 CxList DuplicateTokenParams = All.GetParameters(DuplicateToken);   //all parameters of DuplicateToken method
 CxList DuplicateTokenParamsLP = DuplicateTokenParams.DataInfluencedBy(LogonParams); //parameters of DuplicateToken method that influenced by input
-DuplicateTokenParams = DuplicateTokenParams.GetStartAndEndNodes(CxList.GetStartEndNodesType.EndNodesOnly);
+DuplicateTokenParams = DuplicateTokenParams.GetLastNodesInPath();

 CxList prms = All.NewCxList();

@@ -39,5 +39,8 @@
        }
    }
 }
-result.Add(Impersonate.DataInfluencedBy(prms));
-result.Add(All.FindByShortName("CreateProcessWithLogonW", false).DataInfluencedBy(Inputs));
+result.Add(
+   Impersonate.DataInfluencedBy(prms),
+   All.FindByShortName("CreateProcessWithLogonW", false).DataInfluencedBy(Inputs));
+
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Low_Visibility / Information_Exposure_Through_an_Error_Message

Code changes

```
---
+++
@@ -1,29 +1,30 @@
+CxList ctch = Find_Catch();
+CxList interactiveOutputs = Find_Interactive_Outputs();
+
 if(!All.isWebApplication)
 {
-   CxList ctch = All.FindByType(typeof(Catch));
-
    CxList exc = All.FindAllReferences(ctch) - ctch;
    exc.Add(All.FindByName("*Server.GetLastError*", false));
-
-   result = Find_Interactive_Outputs().DataInfluencedBy(exc);
+
+   result = interactiveOutputs.DataInfluencedBy(exc);
 }
 else
 {
-   CxList main_decl = (All.FindByType(typeof(MethodDecl))).FindByName("*.Main", false).FindByFieldAttributes(Modifiers.Public | Modifiers.Static);
+   CxList main_decl = Find_MethodDecls().FindByName("*.Main", false)
+       .FindByFieldAttributes(Modifiers.Public | Modifiers.Static);

-   CxList classes_with_main = All.GetClass(main_decl);
-
-   CxList ctch = All.FindByType(typeof(Catch));
+   CxList classes_with_main = All.GetClass(main_decl);
+
    CxList class_of_ctch_not_with_main = (All - classes_with_main).GetClass(ctch);

    ctch = ctch.GetByAncs(class_of_ctch_not_with_main);

-   CxList class_not_with_main = All.FindByType(typeof(ClassDecl)) - classes_with_main;
+   CxList class_not_with_main = Find_ClassDecl() - classes_with_main;
    class_not_with_main = All.GetByAncs(class_not_with_main);

    CxList exc = All.FindAllReferences(ctch) - ctch;
    exc.Add(class_not_with_main.FindByName("*Server.GetLastError*", false));
    exc.Add(class_not_with_main.FindByName("*InnerException*", false));

-   result = Find_Interactive_Outputs().DataInfluencedBy(exc);
+   result = interactiveOutputs.DataInfluencedBy(exc);
 }
```

VbNet / VbNet_Low_Visibility / Information_Leak_Through_Persistent_Cookies

Code changes

```
---
+++
@@ -3,4 +3,4 @@
 CxList cookie = All.FindByName("*Response.Cookies*", false);
 cookie.Add(All.FindByMemberAccess("HttpResponse.Cookies*", false));

-result = cookie.InfluencedBy(psw);
+result = cookie.InfluencedBy(psw).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Low_Visibility / Insufficiently_Protected_Credentials

Code changes

```
---
+++
@@ -5,4 +5,4 @@
 CxList DB = All.FindByName("*db*",false);
 DB.Add(Find_DB_Out());

-result = psw.DataInfluencedBy(DB);
+result = psw.DataInfluencedBy(DB).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Low_Visibility / Leaving_Temporary_Files

Code changes

```
---
+++
@@ -1,5 +1,7 @@
 CxList tmpFiles = All.FindByName("Path.GetTempFileName", false);
+
 tmpFiles = tmpFiles.DataInfluencingOn(Find_IO());
+
 CxList delete = All.FindByName("File.Delete", false);

 foreach(CxList curTmpFile in tmpFiles)
```

VbNet / VbNet_Low_Visibility / Log_Forging

Code changes

```
---
+++
@@ -3,4 +3,4 @@

 CxList sanitize = Find_Integers();

-result = Log.InfluencedByAndNotSanitized(Inputs, sanitize);
+result = Log.InfluencedByAndNotSanitized(Inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Low_Visibility / Open_Redirect

Code changes

```
---
+++
```

```
@@ -1,15 +1,13 @@
 CxList redirect = All.FindByMemberAccess("HttpResponse.Redirect", false);
 redirect.Add(All.FindByName("*Response.Redirect", false));

-CxList inputs = All.FindByMemberAccess("*HttpRequest.QueryString_*", false);
-inputs.Add(All.FindByMemberAccess("*HttpRequest.QueryString.item", false));
-inputs.Add(All.FindByName("*Request.QueryString_*", false));
-inputs.Add(All.FindByName("*Request.QueryString.Item", false));
-inputs.Add(All.FindByName("*Request.Item", false));
-inputs.Add(All.FindByShortName("Request", false).FindByFathers(All.FindByType(typeof(IndexerRef))));
+CxList inputs = All.FindByMemberAccesses(new [] {"*HttpRequest.QueryString_*", "*HttpRequest.QueryString.item"}, false);
+inputs.Add(
+   All.FindByNames(new [] {"*Request.QueryString_*", "*Request.QueryString.Item", "*Request.Item"}, false),
+   All.FindByShortName("Request", false).FindByFathers(Find_IndexerRefs()));

 inputs -= inputs.DataInfluencingOn(inputs);

 CxList sanitize = Find_Integers();

-result = redirect.InfluencedByAndNotSanitized(inputs, sanitize);
+result = redirect.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Low_Visibility / Session_Clearing_Problems

Code changes

```
---
+++
@@ -1,6 +1,4 @@
-CxList session = All.FindByShortName("*Session_User*", false);
-session.Add(All.FindByShortName("*Session_cust*", false));
-session.Add(All.FindByShortName("*Session_id*", false));
+CxList session = All.FindByShortNames(new List<string>{"*Session_User*","*Session_cust*","*Session_id*"}, false);

 if(session.Count != 0)
 {
@@ -12,11 +10,11 @@
     clear.Add(All.FindByMemberAccess("*FormsAuthentication.SignOut", false));

     CxList a = All.FindByAssignmentSide(CxList.AssignmentSide.Left) * session;
-    CxList b = All.FindByAssignmentSide(CxList.AssignmentSide.Right) * (emptyString + zero);
+    CxList b = All.FindByAssignmentSide(CxList.AssignmentSide.Right) * All.NewCxList(emptyString, zero);

     CxList c = a.GetFathers() * b.GetFathers();

-    if((c + clear).Count == 0)
+    if(All.NewCxList(c,clear).Count == 0)
     {
         result = session;
     }
```

VbNet / VbNet_Low_Visibility / Stored_Code_Injection

Code changes

```
---
+++
@@ -1,19 +1,14 @@
 CxList methods = Find_Methods();

-CxList CodeProvider = All.FindByMemberAccess("CSharpCodeProvider.*", false);
-CodeProvider.Add(All.FindByMemberAccess("VBCodeProvider.*", false));
-CodeProvider.Add(All.FindByMemberAccess("MethodInfo.*", false));
-CodeProvider.Add(All.FindByMemberAccess("JScriptCodeProvider.*", false));
-CodeProvider.Add(All.FindByMemberAccess("CodeDomProvider.*", false));
-
-//find only codeCompilers (clean anrelevant methods)
-List < string > methCodeCompilers = new List<string> {"CompileAssemblyFrom*", "Parse"};
-CxList codeCompilers = CodeProvider.FindByShortNames(methCodeCompilers, false);
+//find only codeCompilers (clean and relevant methods)
+CxList codeCompilers = methods.FindByMemberAccesses(
+   new [] {"CSharpCodeProvider", "VBCodeProvider", "MethodInfo", "JScriptCodeProvider", "CodeDomProvider"},
+   new [] {"CompileAssemblyFrom*", "Parse"},
+   false);

 CxList declarators = Find_Declarators();
-CxList inputs = Find_DB_Out();
-inputs.Add(Find_Read());
+CxList inputs = All.NewCxList(Find_DB_Out(),Find_Read());
 inputs.Add(All.FindAllReferences(inputs.GetAssignee().FindByType("*Reader", false)));
 inputs.Add(declarators.FindDefinition(inputs.FindByType("DataSet", false)));

-result = inputs.DataInfluencingOn(codeCompilers);
+result = inputs.DataInfluencingOn(codeCompilers).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Low_Visibility / Thread_Safety_Issue

Code changes

```
---
+++
@@ -6,8 +6,9 @@
 CxList statics = no_logs.FindAllReferences(no_logs.FindByFieldAttributes(Modifiers.Static) -
                                            no_logs.FindByFieldAttributes(Modifiers.Readonly));

-statics = statics - no_logs.FindByType(typeof(MethodInvokeExpr));
+statics = statics - no_logs.FindByType<MethodInvokeExpr>();

 CxList EventArgs =  All.FindByType("*CommandEventArgs", false);

-result = (EventArgs + Find_Interactive_Inputs()).DataInfluencingOn(statics);
+result = All.NewCxList(EventArgs, Find_Interactive_Inputs()).DataInfluencingOn(statics)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Low_Visibility / Trust_Boundary_Violation_in_Session_Variables

Code changes

```
---
+++
@@ -3,4 +3,4 @@
 sanitize.Add(All.FindByShortName("Session", false));
 CxList input = Find_Inputs();

-result = setAttr.InfluencedByAndNotSanitized(input, sanitize);
+result = setAttr.InfluencedByAndNotSanitized(input, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Low_Visibility / URL_Canonicalization_Issue

Code changes

```
---
+++
@@ -1,9 +1,9 @@
 CxList inputs = All.FindByMemberAccess("*HttpRequest.RawUrl", false);
 inputs.Add(All.FindByName("*Request.RawUrl", false));

-CxList binaryExpr = All.FindByType(typeof(BinaryExpr));
+CxList binaryExpr = Find_BinaryExpr();
 CxList sanitize = All.FindByName("*Server.UrlDecode", false);

-CxList bin = binaryExpr.InfluencedByAndNotSanitized(inputs, sanitize);
+CxList bin = binaryExpr.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);

 result = bin.ControlInfluencingOn(All);
```

VbNet / VbNet_Low_Visibility / Use_of_Broken_or_Risky_Cryptographic_Algorithm

Code changes

```
---
+++
@@ -1,6 +1,9 @@
 CxList methods = Find_Methods();

 result.Add(All.FindByType("MD5", false).GetMembersOfTarget().FindByShortName("ComputeHash", false));
+
 result.Add(methods.FindByMemberAccess("MD5CryptoServiceProvider.ComputeHash", false));
+
 result.Add(All.FindByType("SHA1", false).GetMembersOfTarget().FindByShortName("ComputeHash", false));
+
 result.Add(methods.FindByMemberAccess("SHA1CryptoServiceProvider.ComputeHash", false));
```

VbNet / VbNet_Medium_Threat / Buffer_Overflow

Code changes

```
---
+++
@@ -1,8 +1,8 @@
```

```
 CxList Inputs = Find_Interactive_Inputs();
 CxList Length = All.FindByName("*.Length", false).DataInfluencedBy(Inputs);


-CxList DllImport = All.FindByName("*DllImport*", false).FindByType(typeof(CustomAttribute));
-CxList ExternalMethod = DllImport.GetFathers().FindByType(typeof(MethodDecl));
+CxList DllImport = Find_CustomAttribute().FindByName("*DllImport*", false);
+CxList ExternalMethod = DllImport.GetFathers().FindByType<MethodDecl>();
 ExternalMethod = All.FindAllReferences(ExternalMethod);


 CxList ExternalMethodParams = All.GetParameters(ExternalMethod);
@@ -11,4 +11,5 @@
 CxList Unsafe = All.GetByMethod(All.FindByFieldAttributes(Modifiers.Unsafe));
 Unsafe  = Unsafe  - Unsafe.InfluencedBy(Length);


-result = (Unsafe + ExternalMethodParams).DataInfluencedBy(Inputs);
+result = All.NewCxList(Unsafe, ExternalMethodParams).DataInfluencedBy(Inputs)
+   .ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Medium_Threat / CGI_XSS

Code changes

```
---
+++
@@ -5,8 +5,7 @@
 //Get All Params and filter out typeof Param and StringLiteral
 CxList par = All.GetParameters(consoleOut);
-par -= par.FindByType(typeof(Param));
-par -= par.FindByType(typeof(StringLiteral));
+par -= par.FindByTypes(typeof(Param), typeof(StringLiteral));


 CxList outputs = All.NewCxList();


@@ -29,11 +28,12 @@
 //Add the results that are both input and output but are not sanitized
 foreach (CxList output in outputs.GetCxListByPath())
 {  //Get the start node that is also an input
-   CxList pathStart = output.GetStartAndEndNodes(CxList.GetStartEndNodesType.StartNodesOnly) * inputs;
-   CxList pathEnd = output.GetStartAndEndNodes(CxList.GetStartEndNodesType.EndNodesOnly); //Get end node
+   CxList pathStart = output.GetFirstNodesInPath() * inputs;
+   CxList pathEnd = output.GetLastNodesInPath(); //Get end node
     //If the start node and end node are not a sanitizer
-   if ((pathStart + pathEnd - sanitized).Count == 2)
+   if ((All.NewCxList(pathStart, pathEnd) - sanitized).Count == 2)
     {   //If the path is not sanitized, then add to result
         result.Add(output);
     }
 }
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Medium_Threat / CSRF

Code changes

```
---
+++
@@ -16,9 +16,12 @@
    CxList db = Find_DB_In();
    CxList strings = Find_Strings();

-   CxList write = strings.FindByNames(new string [] {"*update*", "*delete*", "*insert*"}, StringComparison.OrdinalIgnoreCase);
+   CxList write = strings.FindByNames(
+       new string [] {"*update*", "*delete*", "*insert*"},
+       StringComparison.OrdinalIgnoreCase);

    result = db.InfluencedByAndNotSanitized(write, sanitizers).InfluencedByAndNotSanitized(requests, sanitizers);

    result.Add(Find_ASP_MVC_CSRF());
+   result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
 }
```

VbNet / VbNet_Medium_Threat / Data_Filter_Injection

Code changes

```
---
+++
@@ -4,4 +4,4 @@
 CxList inputs = Find_Interactive_Inputs();
 CxList sanitized = Find_Sanitize();

-result = All.FindSQLInjections(inputs, db, sanitized);
+result = All.FindSQLInjections(inputs, db, sanitized).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Medium_Threat / DoS_by_Sleep

Code changes

```
---
+++
@@ -1,3 +1,3 @@
 CxList Inputs = Find_Interactive_Inputs();
 CxList sleep = All.FindByName("*Thread.Sleep", false);
-result = sleep.DataInfluencedBy(Inputs);
+result = sleep.DataInfluencedBy(Inputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Medium_Threat / Hardcoded_password_in_Connection_String

Code changes

```
---
+++
@@ -2,12 +2,12 @@
 CxList psw = Find_Password_Strings();

 // Find all the connections
```

```
-CxList createExpressions = All.FindByType(typeof(ObjectCreateExpr));
-CxList openConnection = createExpressions.FindByShortName("*Connection", false) +
-   createExpressions.FindByShortName("*DataContext", false) +
-   All.FindByMemberAccess("Connection.ConnectionString", false);
+CxList createExpressions = Find_ObjectCreations();
+CxList openConnection = All.NewCxList(
+   createExpressions.FindByShortNames(new [] {"*Connection", "*DataContext"}, false),
+   All.FindByMemberAccess("Connection.ConnectionString", false));

-CxList sanitizers = Find_Same_Value_Sanitizers_For_Hardcoded_Password(openConnection + psw);
+CxList sanitizers = Find_Same_Value_Sanitizers_For_Hardcoded_Password(All.NewCxList(openConnection, psw));

 // Return the connections influenced by "passworded" strings.
 // Notice that since it's an ObjectCreateExpr ("new()"), only its parameters
@@ -17,6 +17,7 @@
 // Add connection strings that contain a password in their initialization
 CxList getConnection = All.FindByMemberAccess("DriverManager.GetConnection", false);
 CxList connectionStrings = getConnection.DataInfluencedBy(All.GetParameters(getConnection, 2)
-   .FindByType(typeof(StringLiteral)));
+   .FindByType<StringLiteral>());

 result.Add(connectionStrings);
+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Medium_Threat / SQL_Injection_Evasion_Attack

Code changes

```
---
+++
@@ -4,4 +4,4 @@
 CxList sanitize = Find_Sanitize();
 CxList db = Find_SQL_DB_In();

-result = db.InfluencedByAndNotSanitized(decode, sanitize);
+result = db.InfluencedByAndNotSanitized(decode, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Medium_Threat / Stored_Command_Injection

Code changes

```
---
+++
@@ -1,6 +1,7 @@
 CxList inputs = Find_Read();
+
 inputs.Add(Find_DB_Out());

 CxList exec = Find_Command_Execution();

-result = exec.DataInfluencedBy(inputs);
+result = exec.DataInfluencedBy(inputs).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Medium_Threat / Stored_LDAP_Injection

Code changes

```
---

+++

@@ -3,4 +3,4 @@

 CxList dirs = Find_LDAP_Output();

 CxList sanitize = Find_LDAP_Sanitize();


-result = dirs.InfluencedByAndNotSanitized(inputs, sanitize);

+result = dirs.InfluencedByAndNotSanitized(inputs, sanitize).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Medium_Threat / Stored_XPath_Injection

Code changes

```
---

+++

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

 inputs.Add(Find_DB_Out());

 CxList sanitized = Find_Sanitize();


-result = XPath.InfluencedByAndNotSanitized(inputs, sanitized);

+result = XPath.InfluencedByAndNotSanitized(inputs, sanitized).ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```

VbNet / VbNet_Medium_Threat / Use_of_Hard_coded_Cryptographic_Key

Code changes

```
---

+++

@@ -3,12 +3,15 @@

 CxList keys = All.FindByShortName("*enc*", false).FindByShortName("*key*", false);


 CxList key_in_lSide = keys.FindByAssignmentSide(CxList.AssignmentSide.Left);

-CxList strLiterals = All.FindByType(typeof(PrimitiveExpr)) - emptyString - NULL;

-CxList lit_in_rSide = strLiterals.FindByAssignmentSide(CxList.AssignmentSide.Right);

+CxList emptyStrAndNull = All.NewCxList(emptyString, NULL);

+CxList primitiveExprs = Find_PrimitiveExpr() - emptyStrAndNull;

+CxList lit_in_rSide = primitiveExprs.FindByAssignmentSide(CxList.AssignmentSide.Right);


 result = key_in_lSide.FindByFathers(key_in_lSide.GetFathers() * lit_in_rSide.GetFathers());

-CxList key_in_Field = All.GetByAncs(All.FindByType(typeof(FieldDecl)) + All.FindByType(typeof(ConstantDecl)))*keys;

+CxList key_in_Field = All.GetByAncs(All.NewCxList(Find_FieldDecls(), Find_ConstantDecl())) * keys;


-CxList sanitizers = Find_Same_Value_Sanitizers_For_Hardcoded_Password(key_in_Field + strLiterals);

+CxList sanitizers = Find_Same_Value_Sanitizers_For_Hardcoded_Password(All.NewCxList(key_in_Field, primitiveExprs));


-result.Add(key_in_Field.InfluencedByAndNotSanitized(strLiterals, sanitizers));

+result.Add(key_in_Field.InfluencedByAndNotSanitized(primitiveExprs, sanitizers));

+

+result = result.ReduceFlow(CxList.ReduceFlowType.ReduceBigFlow);
```