

SolarWinds Post-Compromise Hunting with Azure Sentinel

By shainw

Published: 2020-12-16 · Archived: 2026-04-05 13:26:22 UTC

```
"}}, "componentScriptGroups({\"componentId\": \"custom.widget.SocialSharing\"}):  
{ \"__typename\": \"ComponentScriptGroups\", \"scriptGroups\":  
{ \"__typename\": \"ComponentScriptGroupsDefinition\", \"afterInteractive\":  
{ \"__typename\": \"PageScriptGroupDefinition\", \"group\": \"AFTER_INTERACTIVE\", \"scriptIds\": [], \"lazyOnLoad\":  
{ \"__typename\": \"PageScriptGroupDefinition\", \"group\": \"LAZY_ON_LOAD\", \"scriptIds\": [], \"componentScripts\":  
[], \"component({\"componentId\": \"custom.widget.MicrosoftFooter\"}):  
{ \"__typename\": \"Component\", \"render({\"context\": {\"component\": {\"entities\": [], \"props\": {}}, \"page\": {\"entities\":  
[\"message:1995095\"], \"name\": \"BlogMessagePage\", \"props\":  
{}, \"url\": \"https://techcommunity.microsoft.com/blog/microsoftsentinelblog/solarwinds-post-compromise-hunting-with-  
azure-sentinel/1995095\"}}}: { \"__typename\": \"ComponentRenderResult\", \"html\": \"  
\"}}, \"componentScriptGroups({\"componentId\": \"custom.widget.MicrosoftFooter\"}):  
{ \"__typename\": \"ComponentScriptGroups\", \"scriptGroups\":  
{ \"__typename\": \"ComponentScriptGroupsDefinition\", \"afterInteractive\":  
{ \"__typename\": \"PageScriptGroupDefinition\", \"group\": \"AFTER_INTERACTIVE\", \"scriptIds\": [], \"lazyOnLoad\":  
{ \"__typename\": \"PageScriptGroupDefinition\", \"group\": \"LAZY_ON_LOAD\", \"scriptIds\": [], \"componentScripts\":  
[], \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":  
[\"components/community/NavbarDropdownToggle\"]: { \"__ref\": \"CachedAsset:text:en_US-  
components/community/NavbarDropdownToggle-  
1775111750899\"}}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":  
[\"components/messages/MessageCoverImage\"]: { \"__ref\": \"CachedAsset:text:en_US-  
components/messages/MessageCoverImage-  
1775111750899\"}}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":  
[\"shared/client/components/nodes/NodeTitle\"]: { \"__ref\": \"CachedAsset:text:en_US-  
shared/client/components/nodes/NodeTitle-  
1775111750899\"}}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":  
[\"components/messages/MessageTimeToRead\"]: { \"__ref\": \"CachedAsset:text:en_US-  
components/messages/MessageTimeToRead-  
1775111750899\"}}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":  
[\"components/messages/MessageSubject\"]: { \"__ref\": \"CachedAsset:text:en_US-components/messages/MessageSubject-  
1775111750899\"}}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":  
[\"components/users/UserLink\"]: { \"__ref\": \"CachedAsset:text:en_US-components/users/UserLink-  
1775111750899\"}}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":  
[\"shared/client/components/users/UserRank\"]: { \"__ref\": \"CachedAsset:text:en_US-  
shared/client/components/users/UserRank-  
1775111750899\"}}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":  
[\"components/messages/MessageTime\"]: { \"__ref\": \"CachedAsset:text:en_US-components/messages/MessageTime-  
1775111750899\"}}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":  
[\"components/messages/MessageBody\"]: { \"__ref\": \"CachedAsset:text:en_US-components/messages/MessageBody-  
1775111750899\"}}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":  
[\"components/messages/MessageCustomFields\"]: { \"__ref\": \"CachedAsset:text:en_US-  
components/messages/MessageCustomFields-  
1775111750899\"}}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":  
[\"components/messages/MessageRevision\"]: { \"__ref\": \"CachedAsset:text:en_US-  
components/messages/MessageRevision-  
1775111750899\"}}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":  
[\"shared/client/components/common/QueryHandler\"]: { \"__ref\": \"CachedAsset:text:en_US-  
shared/client/components/common/QueryHandler-  
1775111750899\"}}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":  
[\"components/tags/TagList\"]: { \"__ref\": \"CachedAsset:text:en_US-components/tags/TagList-  
1775111750899\"}}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":  
[\"components/messages/MessageReplyButton\"]: { \"__ref\": \"CachedAsset:text:en_US-  
components/messages/MessageReplyButton-  
1775111750899\"}}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":  
[\"components/messages/MessageAuthorBio\"]: { \"__ref\": \"CachedAsset:text:en_US-  
components/messages/MessageAuthorBio-  
1775111750899\"}}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":  
[\"shared/client/components/users/UserAvatar\"]: { \"__ref\": \"CachedAsset:text:en_US-
```

```
shared/client/components/users/UserAvatar-
1775111750899"}, "cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":
[\"shared/client/components/ranks/UserRankLabel\"]}): {\"__ref\": \"CachedAsset:text:en_US-
shared/client/components/ranks/UserRankLabel-
1775111750899\"}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":
[\"components/tags/TagView/TagViewChip\"]}): {\"__ref\": \"CachedAsset:text:en_US-
components/tags/TagView/TagViewChip-
1775111750899\"}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":
[\"components/users/UserRegistrationDate\"]}): {\"__ref\": \"CachedAsset:text:en_US-
components/users/UserRegistrationDate-
1775111750899\"}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":
[\"shared/client/components/nodes/NodeAvatar\"]}): {\"__ref\": \"CachedAsset:text:en_US-
shared/client/components/nodes/NodeAvatar-
1775111750899\"}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":
[\"shared/client/components/nodes/NodeDescription\"]}): {\"__ref\": \"CachedAsset:text:en_US-
shared/client/components/nodes/NodeDescription-
1775111750899\"}, \"cachedText({\"lastModified\": \"1775111750899\", \"locale\": \"en-US\", \"namespaces\":
[\"shared/client/components/nodes/NodeIcon\"]}): {\"__ref\": \"CachedAsset:text:en_US-
shared/client/components/nodes/NodeIcon-1775111750899\"}}, \"Theme:customTheme1\":
{ \"__typename\": \"Theme\", \"id\": \"customTheme1\", \"User:user:-1\":
{ \"__typename\": \"User\", \"id\": \"user:-1\", \"entityType\": \"USER\", \"eventPath\": \"community:gxucf89792/user:-1\", \"uid\": \"-1\", \"login\": \"Deleted\", \"email\": \"\", \"avatar\":
{ \"__typename\": \"RegistrationData\", \"status\": \"ANONYMOUS\", \"registrationTime\": null, \"confirmEmailStatus\": false, \"registrationAccessLevel\": \"VIEW\", \"ss
[]\", \"ssoId\": null, \"profileSettings\": { \"__typename\": \"ProfileSettings\", \"dateDisplayStyle\":
{ \"__typename\": \"InheritableStringSettingWithPossibleValues\", \"key\": \"layout.friendly_dates_enabled\", \"value\": \"false\", \"localValue\": \"true\", \"possibleValues\"
[\"true\", \"false\"]}, \"dateDisplayFormat\":
{ \"__typename\": \"InheritableStringSetting\", \"key\": \"layout.format_pattern_date\", \"value\": \"MMM dd yyyy\", \"localValue\": \"MM-
dd-yyyy\", \"language\": { \"__typename\": \"InheritableStringSettingWithPossibleValues\", \"key\": \"profile.language\", \"value\": \"en-
US\", \"localValue\": null, \"possibleValues\": [\"en-US\", \"es-ES\"]}, \"repliesSortOrder\":
{ \"__typename\": \"InheritableStringSettingWithPossibleValues\", \"key\": \"config.user_replies_sort_order\", \"value\": \"DEFAULT\", \"localValue\": \"DEFAULT\", \"pc
[\"DEFAULT\", \"LIKES\", \"PUBLISH_TIME\", \"REVERSE_PUBLISH_TIME\"]}, \"deleted\": false, \"CachedAsset:pages-
1775111737667\": { \"__typename\": \"CachedAsset\", \"id\": \"pages-1775111737667\", \"value\":
[ { \"lastUpdatedTime\": 1775111737667, \"localOverride\": null, \"page\":
{ \"id\": \"BlogViewAllPostsPage\", \"type\": \"BLOG\", \"urlPath\": \"/category/:categoryId/blog/:boardId/all-
posts(/:after|/:before)?\", \"__typename\": \"PageDescriptor\"}, \"__typename\": \"PageResource\"},
{ \"lastUpdatedTime\": 1775111737667, \"localOverride\": null, \"page\":
{ \"id\": \"CasePortalPage\", \"type\": \"CASE_PORTAL\", \"urlPath\": \"/caseportal\", \"__typename\": \"PageDescriptor\"}, \"__typename\": \"PageResource\"},
{ \"lastUpdatedTime\": 1775111737667, \"localOverride\": null, \"page\":
{ \"id\": \"CreateGroupHubPage\", \"type\": \"GROUP_HUB\", \"urlPath\": \"/groups/create\", \"__typename\": \"PageDescriptor\"}, \"__typename\": \"PageResource\"},
{ \"lastUpdatedTime\": 1775111737667, \"localOverride\": null, \"page\":
{ \"id\": \"CaseViewPage\", \"type\": \"CASE_DETAILS\", \"urlPath\": \"/case/:caseId/:caseNumber\", \"__typename\": \"PageDescriptor\"}, \"__typename\": \"PageResource\"},
{ \"lastUpdatedTime\": 1775111737667, \"localOverride\": null, \"page\":
{ \"id\": \"InboxPage\", \"type\": \"COMMUNITY\", \"urlPath\": \"/inbox\", \"__typename\": \"PageDescriptor\"}, \"__typename\": \"PageResource\"},
{ \"lastUpdatedTime\": 1775111737667, \"localOverride\": null, \"page\":
{ \"id\": \"HelpFAQPage\", \"type\": \"COMMUNITY\", \"urlPath\": \"/help\", \"__typename\": \"PageDescriptor\"}, \"__typename\": \"PageResource\"},
{ \"lastUpdatedTime\": 1775111737667, \"localOverride\": null, \"page\":
{ \"id\": \"IdeaMessagePage\", \"type\": \"IDEA_POST\", \"urlPath\": \"/idea/:boardId/:messageSubject/:messageId\", \"__typename\": \"PageDescriptor\"}, \"__typename\":
{ \"lastUpdatedTime\": 1775111737667, \"localOverride\": null, \"page\":
{ \"id\": \"IdeaViewAllIdeasPage\", \"type\": \"IDEA\", \"urlPath\": \"/category/:categoryId/ideas/:boardId/all-
ideas(/:after|/:before)?\", \"__typename\": \"PageDescriptor\"}, \"__typename\": \"PageResource\"},
{ \"lastUpdatedTime\": 1775111737667, \"localOverride\": null, \"page\":
{ \"id\": \"LoginPage\", \"type\": \"USER\", \"urlPath\": \"/signin\", \"__typename\": \"PageDescriptor\"}, \"__typename\": \"PageResource\"},
{ \"lastUpdatedTime\": 1775111737667, \"localOverride\": null, \"page\":
{ \"id\": \"WorkstreamsPage\", \"type\": \"COMMUNITY\", \"urlPath\": \"/workstreams\", \"__typename\": \"PageDescriptor\"}, \"__typename\": \"PageResource\"},
{ \"lastUpdatedTime\": 1775111737667, \"localOverride\": null, \"page\":
{ \"id\": \"BlogPostPage\", \"type\": \"BLOG\", \"urlPath\": \"/category/:categoryId/blogs/:boardId/create\", \"__typename\": \"PageDescriptor\"}, \"__typename\": \"PageRes
{ \"lastUpdatedTime\": 1775111737667, \"localOverride\": null, \"page\":
{ \"id\": \"UserBlogPermissions.Page\", \"type\": \"COMMUNITY\", \"urlPath\": \"/c/user-blog-
permissions/page\", \"__typename\": \"PageDescriptor\"}, \"__typename\": \"PageResource\"},
{ \"lastUpdatedTime\": 1775111737667, \"localOverride\": null, \"page\":
{ \"id\": \"ThemeEditorPage\", \"type\": \"COMMUNITY\", \"urlPath\": \"/designer/themes\", \"__typename\": \"PageDescriptor\"}, \"__typename\": \"PageResource\"},
{ \"lastUpdatedTime\": 1775111737667, \"localOverride\": null, \"page\":
{ \"id\": \"TkbViewAllArticlesPage\", \"type\": \"TKB\", \"urlPath\": \"/category/:categoryId/kb/:boardId/all-
articles(/:after|/:before)?\", \"__typename\": \"PageDescriptor\"}, \"__typename\": \"PageResource\"},
{ \"lastUpdatedTime\": 1730819800000, \"localOverride\": null, \"page\":
```

```
{"id": "AllEvents", "type": "CUSTOM", "urlPath": "/Events", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "OccasionEditPage", "type": "EVENT", "urlPath": "/event/:boardId/:messageSubject/:messageId/edit", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "OAuthAuthorizationAllowPage", "type": "USER", "urlPath": "/auth/authorize/allow", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "PageEditorPage", "type": "COMMUNITY", "urlPath": "/designer/pages", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "PostPage", "type": "COMMUNITY", "urlPath": "/category/:categoryId/:boardId/create", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "CreateUserGroupPage", "type": "COMMUNITY", "urlPath": "/c/create-user-group/page", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "ForumBoardPage", "type": "FORUM", "urlPath": "/category/:categoryId/discussions/:boardId", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "TkbBoardPage", "type": "TKB", "urlPath": "/category/:categoryId/kb/:boardId", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "EventPostPage", "type": "EVENT", "urlPath": "/category/:categoryId/events/:boardId/create", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "UserBadgesPage", "type": "COMMUNITY", "urlPath": "/users/:login/userId/badges", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "GroupHubMembershipAction", "type": "GROUP_HUB", "urlPath": "/membership/join/:nodeId/membershipType", "__typename": "PageDescriptor"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "MaintenancePage", "type": "COMMUNITY", "urlPath": "/maintenance", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "IdeaReplyPage", "type": "IDEA_REPLY", "urlPath": "/idea/:boardId/:messageSubject/:messageId/comments/:replyId", "__typename": "PageDescriptor"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "UserSettingsPage", "type": "USER", "urlPath": "/mysettings/userSettingsTab", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "GroupHubsPage", "type": "GROUP_HUB", "urlPath": "/groups", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "ForumPostPage", "type": "FORUM", "urlPath": "/category/:categoryId/discussions/:boardId/create", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "OccasionRsvpActionPage", "type": "OCCASION", "urlPath": "/event/:boardId/:messageSubject/:messageId/rsvp/:responseType", "__typename": "PageDescriptor"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "VerifyUserEmailPage", "type": "USER", "urlPath": "/verifyemail/userId/verifyEmailToken", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "AllOccasionsPage", "type": "OCCASION", "urlPath": "/category/:categoryId/events/:boardId/all-events(/:after/:before)?", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "EventBoardPage", "type": "EVENT", "urlPath": "/category/:categoryId/events/:boardId", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "TkbReplyPage", "type": "TKB_REPLY", "urlPath": "/kb/:boardId/:messageSubject/:messageId/comments/:replyId", "__typename": "PageDescriptor"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "IdeaBoardPage", "type": "IDEA", "urlPath": "/category/:categoryId/ideas/:boardId", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "CommunityGuideLinesPage", "type": "COMMUNITY", "urlPath": "/communityguidelines", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "CaseCreatePage", "type": "SALESFORCE_CASE_CREATION", "urlPath": "/caseportal/create", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "TkbEditPage", "type": "TKB", "urlPath": "/kb/:boardId/:messageSubject/:messageId/edit", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "ForgotPasswordPage", "type": "USER", "urlPath": "/forgotpassword", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "IdeaEditPage", "type": "IDEA", "urlPath": "/idea/:boardId/:messageSubject/:messageId/edit", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "TagPage", "type": "COMMUNITY", "urlPath": "/tag/:tagName", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "BlogBoardPage", "type": "BLOG", "urlPath": "/category/:categoryId/blog/:boardId", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "OccasionMessagePage", "type": "OCCASION_TOPIC", "urlPath": "/event/:boardId/:messageSubject/:messageId", "__typename": "PageDescriptor"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page": {"id": "ManageContentPage", "type": "COMMUNITY", "urlPath": "/managecontent", "__typename": "PageDescriptor"}, {"__typename": "PageResource"}, {"lastUpdatedTime": 1775111737667, "localOverride": null, "page":
```

```
{ "id": "ClosedMembershipNodeNonMembersPage", "type": "GROUP_HUB", "urlPath": "/closedgroup:/groupHubId", "__typename": "PageDescriptor", "__type": { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "CommunityPage", "type": "COMMUNITY", "urlPath": "/", "__typename": "PageDescriptor", "__typename": "PageResource", { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "ForumMessagePage", "type": "FORUM_TOPIC", "urlPath": "/discussions:/boardId:/messageSubject:/messageId", "__typename": "PageDescriptor", " __type": { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "IdeaPostPage", "type": "IDEA", "urlPath": "/category:/categoryId/ideas:/boardId/create", "__typename": "PageDescriptor", "__typename": "PageResou { "lastUpdatedTime": 1730819800000, "localOverride": null, "page": { "id": "CommunityHub.Page", "type": "CUSTOM", "urlPath": "/Directory", "__typename": "PageDescriptor", "__typename": "PageResource", { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "BlogMessagePage", "type": "BLOG_ARTICLE", "urlPath": "/blog:/boardId:/messageSubject:/messageId", "__typename": "PageDescriptor", "__typeper { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "RegistrationPage", "type": "USER", "urlPath": "/register", "__typename": "PageDescriptor", "__typename": "PageResource", { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "EditGroupHubPage", "type": "GROUP_HUB", "urlPath": "/group:/groupHubId/edit", "__typename": "PageDescriptor", "__typename": "PageResource { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "ForumEditPage", "type": "FORUM", "urlPath": "/discussions:/boardId:/messageSubject:/messageId/edit", "__typename": "PageDescriptor", "__typena { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "ResetPasswordPage", "type": "USER", "urlPath": "/resetpassword:/userId:/resetPasswordToken", "__typename": "PageDescriptor", "__typename": "Pa { "lastUpdatedTime": 1730819800000, "localOverride": null, "page": { "id": "AllBlogs.Page", "type": "CUSTOM", "urlPath": "/blogs", "__typename": "PageDescriptor", "__typename": "PageResource", { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "TkbMessagePage", "type": "TKB_ARTICLE", "urlPath": "/kb:/boardId:/messageSubject:/messageId", "__typename": "PageDescriptor", "__typename { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "BlogEditPage", "type": "BLOG", "urlPath": "/blog:/boardId:/messageSubject:/messageId/edit", "__typename": "PageDescriptor", "__typename": "Page { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "ManageUsersPage", "type": "USER", "urlPath": "/users/management:/tab?/:manageUsersTab?", "__typename": "PageDescriptor", "__typename": "PageRes { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "ForumReplyPage", "type": "FORUM_REPLY", "urlPath": "/discussions:/boardId:/messageSubject:/messageId/replies:/replyId", "__typename": "Page { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "PrivacyPolicyPage", "type": "COMMUNITY", "urlPath": "/privacypolicy", "__typename": "PageDescriptor", "__typename": "PageResource", { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "NotificationPage", "type": "COMMUNITY", "urlPath": "/notifications", "__typename": "PageDescriptor", "__typename": "PageResource", { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "UserPage", "type": "USER", "urlPath": "/users:/login:/userId", "__typename": "PageDescriptor", "__typename": "PageResource", { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "HealthCheckPage", "type": "COMMUNITY", "urlPath": "/health", "__typename": "PageDescriptor", "__typename": "PageResource", { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "OccasionReplyPage", "type": "OCCASION_REPLY", "urlPath": "/event:/boardId:/messageSubject:/messageId/comments:/replyId", "__typename": "P { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "ManageMembersPage", "type": "GROUP_HUB", "urlPath": "/group:/groupHubId/management:/tab?", "__typename": "PageDescriptor", "__typename": "P { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "SearchResultsPage", "type": "COMMUNITY", "urlPath": "/search", "__typename": "PageDescriptor", "__typename": "PageResource", { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "BlogReplyPage", "type": "BLOG_REPLY", "urlPath": "/blog:/boardId:/messageSubject:/messageId/replies:/replyId", "__typename": "PageDescriptor" { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "GroupHubPage", "type": "GROUP_HUB", "urlPath": "/group:/groupHubId", "__typename": "PageDescriptor", "__typename": "PageResource", { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "TermsOfServicePage", "type": "COMMUNITY", "urlPath": "/termsofservice", "__typename": "PageDescriptor", "__typename": "PageResource", { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "CategoryPage", "type": "CATEGORY", "urlPath": "/category:/categoryId", "__typename": "PageDescriptor", "__typename": "PageResource", { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "ForumViewAllTopicsPage", "type": "FORUM", "urlPath": "/category:/categoryId/discussions:/boardId/all- { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "TkbPostPage", "type": "TKB", "urlPath": "/category:/categoryId/kbs:/boardId/create", "__typename": "PageDescriptor", "__typename": "PageResource { "lastUpdatedTime": 1775111737667, "localOverride": null, "page": { "id": "GroupHubPostPage", "type": "GROUP_HUB", "urlPath": "/group:/groupHubId:/boardId/create", "__typename": "PageDescriptor", "__typename": "Pa components/context/AppContext/AppContextProvider-0": { "__typename": "CachedAsset", "id": "text:en_US- components/context/AppContext/AppContextProvider-0", "value": { "noCommunity": "Cannot find community", "noUser": "Cannot find current user", "noNode": "Cannot find node with id {nodeId}", "noMessage": "Cannot find message with id {messageId}", "userBanned": "We're sorry, but you have been banned from using this site.", "userBannedReason": "You have been banned for the following reason: {reason}" }, "localOverride": false, "CachedAsset: text:en_US-shared/client/components/common/Loading/LoadingDot-0":
```

```
{ "__typename": "CachedAsset", "id": "text:en_US-shared/client/components/common/Loading/LoadingDot-0", "value":  
{"title": "Loading...", "localOverride": false}, "AssociatedImage":  
{ "url": "https://techcommunity.microsoft.com/t5/s/gxcuf89792/images/cmstNC05WEo0blc"}":  
{ "__typename": "AssociatedImage", "url": "https://techcommunity.microsoft.com/t5/s/gxcuf89792/images/cmstNC05WEo0blc", "height": 512, "width": 512,  
{ "__typename": "Rank", "id": "rank:4", "position": 2, "name": "Microsoft", "color": "333333", "icon": {" __ref": "AssociatedImage":  
{ "url": "https://techcommunity.microsoft.com/t5/s/gxcuf89792/images/cmstNC05WEo0blc"}"}, "rankStyle": "OUTLINE"}, "User": "user:252752":  
{ "__typename": "User", "id": "user:252752", "uid": 252752, "login": "shainw", "deleted": false, "avatar":  
{ "__typename": "UserAvatar", "url": "https://techcommunity.microsoft.com/t5/s/gxcuf89792/images/dS0yNTI3NTItMTU0NzA4aTBCQjIxNjFDMzc2MDI  
{ __ref": "Rank:rank:4"}, "email": "", "messagesCount": 8, "biography": null, "topicsCount": 4, "kudosReceivedCount": 26, "kudosGivenCount": 1, "kudosWeigh  
{ __typename": "RegistrationData", "status": null, "registrationTime": "2018-12-11T12:55:55.825-  
08:00", "confirmEmailStatus": null, "followersCount": null, "solutionsCount": 0, "Category": "category:microsoft-sentinel":  
{ __typename": "Category", "id": "category:microsoft-sentinel", "entityType": "CATEGORY", "displayId": "microsoft-  
sentinel", "nodeType": "category", "depth": 4, "title": "Microsoft Sentinel", "shortTitle": "Microsoft Sentinel", "parent":  
{ __ref": "Category:category:microsoft-security"}}, "Category:category:top":  
{ __typename": "Category", "id": "category:top", "entityType": "CATEGORY", "displayId": "top", "nodeType": "category", "depth": 0, "title": "Top", "shortTitle"  
{ __typename": "Category", "id": "category:communities", "entityType": "CATEGORY", "displayId": "communities", "nodeType": "category", "depth": 1, "pare  
{ __ref": "Category:category:top"}, "title": "Communities", "shortTitle": "Communities"}, "Category:category:products-  
services": {" __typename": "Category", "id": "category:products-services", "entityType": "CATEGORY", "displayId": "products-  
services", "nodeType": "category", "depth": 2, "parent":  
{ __ref": "Category:category:communities"}, "title": "Products", "shortTitle": "Products"}, "Category:category:microsoft-  
security": {" __typename": "Category", "id": "category:microsoft-  
security", "entityType": "CATEGORY", "displayId": "microsoft-security", "nodeType": "category", "depth": 3, "parent":  
{ __ref": "Category:category:products-services"}, "title": "Microsoft Security", "shortTitle": "Microsoft  
Security", "categoryPolicies": {" __typename": "CategoryPolicies", "canReadNode":  
{ __typename": "PolicyResult", "failureReason": null}}, "Blog:board:MicrosoftSentinelBlog":  
{ __typename": "Blog", "id": "board:MicrosoftSentinelBlog", "entityType": "BLOG", "displayId": "MicrosoftSentinelBlog", "nodeType": "board", "depth": 5, "c  
{ __typename": "RepliesProperties", "sortOrder": "REVERSE_PUBLISH_TIME", "repliesFormat": "threaded"}, "tagProperties":  
{ __typename": "TagNodeProperties", "tagsEnabled":  
{ __typename": "PolicyResult", "failureReason": null}}, "requireTags": false, "tagType": "PRESET_ONLY", "description":"
```

Microsoft Sentinel is an industry-leading SIEM & AI-first platform powering agentic defense across the entire security ecosystem.

```
"title": "Microsoft Sentinel Blog", "shortTitle": "Microsoft Sentinel Blog", "parent": {" __ref": "Category:category:microsoft-  
sentinel"}, "ancestors": {" __typename": "CoreNodeConnection", "edges": [{" __typename": "CoreNodeEdge", "node":  
{ __ref": "Community:community:gxcuf89792"}}, {" __typename": "CoreNodeEdge", "node":  
{ __ref": "Category:category:communities"}}, {" __typename": "CoreNodeEdge", "node":  
{ __ref": "Category:category:products-services"}}, {" __typename": "CoreNodeEdge", "node":  
{ __ref": "Category:category:microsoft-security"}}, {" __typename": "CoreNodeEdge", "node":  
{ __ref": "Category:category:microsoft-sentinel"}]}}, "userContext":  
{ __typename": "NodeUserContext", "canAddAttachments": false, "canUpdateNode": false, "canPostMessages": false, "isSubscribed": false, "theme":  
{ __ref": "Theme:customTheme1"}, "boardPolicies": {" __typename": "BoardPolicies", "canViewSpamDashBoard":  
{ __typename": "PolicyResult", "failureReason":  
{ __typename": "FailureReason", "message": "error.lithium.policies.feature.moderation_spam.action.access_spam_quarantine.allowed.accessDenied", "key"  
[]}}, "canArchiveMessage": {" __typename": "PolicyResult", "failureReason":  
{ __typename": "FailureReason", "message": "error.lithium.policies.content_archivals.enable_content_archival_settings.accessDenied", "key": "error.lithium  
[]}}, "canPublishArticleOnCreate": {" __typename": "PolicyResult", "failureReason":  
{ __typename": "FailureReason", "message": "error.lithium.policies.forums.policy_can_publish_on_create_workflow_action.accessDenied", "key": "error.lit  
[]}}, "linkProperties":  
{ __typename": "LinkProperties", "isExternalLinkWarningEnabled": false}, "BlogTopicMessage:message:1995095":  
{ __typename": "BlogTopicMessage", "uid": 1995095, "subject": "SolarWinds Post-Compromise Hunting with Azure  
Sentinel", "id": "message:1995095", "entityType": "BLOG_ARTICLE", "eventPath": "category:microsoft-  
sentinel/category:microsoft-security/category:products-  
services/category:community:gxcuf89792board:MicrosoftSentinelBlog/message:1995095", "revisionNum": 57, "repliesCount": 5, "author":  
{ __ref": "User:user:252752"}, "depth": 0, "hasGivenKudo": false, "board":  
{ __ref": "Blog:board:MicrosoftSentinelBlog"}, "conversation":  
{ __ref": "Conversation:conversation:1995095"}, "messagePolicies":  
{ __typename": "MessagePolicies", "canPublishArticleOnEdit": {" __typename": "PolicyResult", "failureReason":  
{ __typename": "FailureReason", "message": "error.lithium.policies.forums.policy_can_publish_on_edit_workflow_action.accessDenied", "key": "error.lithi  
[]}}, "canModerateSpamMessage": {" __typename": "PolicyResult", "failureReason":  
{ __typename": "FailureReason", "message": "error.lithium.policies.feature.moderation_spam.action.moderate_entity.allowed.accessDenied", "key": "error.li  
[]}}, "canReply": {" __typename": "PolicyResult", "failureReason":  
{ __typename": "FailureReason", "message": "error.lithium.policies.forums.action.message_reply_to_entity.allow.accessDenied", "key": "error.lithium.polic  
[]}}, "canAcceptSolution": {" __typename": "PolicyResult", "failureReason":
```


As shared in Microsoft's technical blog – [Customer Guidance on Recent Nation-state Cyber Attacks](#) - attackers might have compromised the internal build systems or the update distribution systems of SolarWinds Orion software then modified a DLL component in the legitimate software and embedded backdoor code that would allow these attackers to remotely perform commands or deliver additional payloads. Below is a representation of various attack stages which you can also see in [Microsoft Threat Protection \(MTP\) portal](#). Note that if you do not have Microsoft Threat Protection this link will not work for you.

\n\n\n\n

To hunt for similar TTPs used in this attack, a good place to start is to build an inventory of the machines that have SolarWinds Orion components. Organizations might already have a software inventory management system to indicate hosts where the SolarWinds application is installed. Alternatively, Azure Sentinel could be leveraged to run a simple query to gather similar details. Azure Sentinel collects data from multiple different logs that could be used to gather this information. For example, through the recently released [Microsoft 365 Defender connector](#), security teams can now easily ingest Microsoft 365 raw data into Azure Sentinel. Using the ingested data, a simple query like below can be written that will pull the hosts with SolarWinds process running in last 30 days based on Process execution either via host on boarded to Sentinel or on boarded via Microsoft Defender for Endpoints (MDE). The query also leverages the Sysmon logs that a lot of customers are collecting from their environment to surface the machines that have SolarWinds running on them. Similar queries that leverage M365 raw data could also be run from the M365's Advanced hunting portal.

\n\n

[SolarWinds Inventory check query](#).

\n

[Spoiler](#)

\n

\n

let timeframe = 30d;

\n

(union isfuzzy=true

\n

(

\n

SecurityEvent

\n

| where TimeGenerated >= ago(timeframe)

\n

| where EventID == '4688'

\n

| where tolower(NewProcessName) has 'solarwinds'

\n

| extend MachineName = Computer , Process = NewProcessName

\n

|

summarize StartTime = min(TimeGenerated), EndTime = max(TimeGenerated), MachineCount = dcount(MachineName), AccountCount = dcount(Acco

\n

),

\n

(

```
\n
DeviceProcessEvents
\n
| where TimeGenerated >= ago(timeframe)
\n
| where tolower(InitiatingProcessFolderPath) has 'solarwinds'
\n
| extend MachineName = DeviceName , Process = InitiatingProcessFolderPath, Account = AccountName
\n
|
summarize StartTime = min(TimeGenerated), EndTime = max(TimeGenerated), MachineCount = dcount(MachineName), AccountCount = dcount(Acco
\n
),
\n
(
\n
Event
\n
| where TimeGenerated >= ago(timeframe)
\n
| where Source == \"Microsoft-Windows-Sysmon\"
\n
| where EventID == 1
\n
| extend Image = EventDetail.[4].[\"#text\"]
\n
| where tolower(Image) has 'solarwinds'
\n
| extend MachineName = Computer , Process = Image, Account = UserName
\n
|
summarize StartTime = min(TimeGenerated), EndTime = max(TimeGenerated), MachineCount = dcount(MachineName), AccountCount = dcount(Acco
\n
)
\n
)
\n
\n
\n\n
```

On systems where the malicious SolarWinds DLL (SolarWinds.Orion.Core.BusinessLayer.dll) is running, it is known that the attacker used a hardcoded named pipe '583da945-62af-10e8-4902-a8f205c72b2e' to conduct various checks as well as to ensure only one instance of the backdoor was running. The use of named pipes by malware is not uncommon as it provides a mechanism for communication between processes. This activity by the malware can be detected if you are collecting Sysmon (Event Id 17/18) or Security Event Id 5145 in your Azure Sentinel workspace. The [Solorigate Named Pipe detection](#) should not be considered reliable on its own as the creation of just the hardcoded named pipe does not indicate that the malicious code was completely triggered, and the machine beacons out or received additional commands. However, presence of this is definitely suspicious and should warrant further in-depth investigation.

\n

[Spoiler](#)

```
let timeframe = 1d;
(union isfuzzy=true
(Event
| where TimeGenerated >= ago(timeframe)
| where Source == \"Microsoft-Windows-Sysmon\"
| where EventID in (17,18)
| extend EvData = parse_xml(EventData)
| extend EventDetail = EvData.DataItem.EventData.Data
| extend NamedPipe = EventDetail.[5].[\"#text\"]
| extend ProcessDetail = EventDetail.[6].[\"#text\"]
| where NamedPipe contains '583da945-62af-10e8-4902-a8f205c72b2e'
| extend Account = UserName
| project-away EventDetail, EvData
),
(
SecurityEvent
| where TimeGenerated >= ago(timeframe)
| where EventID == '5145'
| where AccessList has '%4418' // presence of CreatePipeInstance value
| where RelativeTargetName contains '583da945-62af-10e8-4902-a8f205c72b2e'
)
)
| extend timestamp = TimeGenerated, AccountCustomEntity = Account, HostCustomEntity = Computer
```

\n\n

Once the adversary acquires an initial foothold on a system thru the SolarWinds process they will have System account level access, the attacker will then attempt to elevate to domain admin level access to the environment. The Microsoft Threat Intelligence Center (MSTIC) team has already delivered multiple queries into Azure Sentinel that identify similar TTPs and many are also available in M365. These methodologies are not specific to just this threat actor or this attack but have been seen in various attack campaigns.

\n

Identifying abnormal logon activities or additions to privileged groups is one way to identify privilege escalation.

\n\n

- \n
 - Checking for [hosts with new logons](#) to identify potential lateral movement by the attacker.
- \n
 - Look for any new [account being created and added to built-in administrators group](#).
- \n
 - Look for any [user account added to privileged built in domain local or global groups](#), including adding accounts to a domain privileged group such as Enterprise Admins, Cert Publishers or DnsAdmins.
- \n
 - Monitor for [rare activity by a high-value account](#) carried out on a system or service.

\n

Related to this attack, in some environments service account credentials had been granted administrative privileges. The above queries can be modified to remove the condition of focusing “User” accounts by commenting the query to include service accounts in the scope where applicable:

\n\n

```
//| where AccountType == \'User\'
```

\n\n

Please see the [Azure Sentinel Github](#) for additional queries and hunting ideas related to Accounts under the Detections and Hunting Queries sections for AuditLogs, and SecurityEvents

\n

Microsoft 365 Defender team has also shared quite a few sample queries for use in their [advanced hunting](#) portal that could be leveraged to detect this part of the attack. Additionally, the logic for many of the Azure Sentinel queries can also be transformed to equivalent queries for Microsoft 365 Defender, that could be run in their Advanced Hunting Portal.

\n

Microsoft 365 Defender has an upcoming complimentary blog that will be updated here once available.

\n\n\n

The next step in the attack was stealing the certificate that signs SAML tokens from the federation server (ADFS) called a Token Signing Cert (TSC). SAML Tokens are basically XML representations of claims. You can read more about ADFS in [What is federation with Azure AD? | Microsoft Docs](#) and SAML at [Azure Single Sign On SAML Protocol - Microsoft identity platform | Microsoft Docs](#). The process is as follows:

\n

- \n
1. A client requests a SAML token from an ADFS Server by authenticating to that server using Windows credentials.
\n
2. The ADFS server issues a SAML token to the client.
\n
3. The SAML token is signed with a certificate associated with the server.
\n
4. The client then presents the SAML token to the application that it needs access to.
\n
5. The signature over the SAML token tells the application that the security token service issued the token and grants access to the client.
\n

\n\n\n

Updated 01/15/2021

\n

The implication of stealing the Token Signing Cert (TSC) is that once the certificate has been acquired, the actor can forge SAML (Security Assertions Markup Language) tokens with whatever claims and lifetime they choose, then sign it with the certificate that has been acquired. Microsoft continues to strongly recommend securing your AD FS (Active Directory Federation Service) TSC because if these TSC's are acquired by a bad actor, this then enables the actor to forge SAML tokens that impersonate highly privileged accounts. There are publicly available pen-testing tools like [ADFSDump](#) and [ADFSpoof](#) that help with extracting required information from the AD FS configuration database to generate the forged security tokens. While we have not confirmed these specific tools were used in this attack, they are useful for simulating the attack behavior or executing a similar attack and therefore, Microsoft has created a high-fidelity detection related to this for M365 Defender:

\n

- \n
• **ADFS private key extraction** which detects ADFS private key extraction patterns from tools such as ADFSDump.
\n

\n

Note: Any M365 Defender alert can be seen in Azure Sentinel Security Alerts or in the M365 security portal.

\n

Updated 01/15/2021

\n

The TTP (tactics, techniques, and procedures) observed in the Solorigate attack is the creation of a legitimate SAML token used to authenticate as any user. One way an attacker could achieve this is by compromising AD FS key material. Microsoft has a new detection for this as stated above and for Azure Sentinel has also created a Windows Event Log based detection that indicates an [ADFS DKM Master Key Export](#). As part of the update for this query to the Azure Sentinel GitHub, there is a [detailed write up](#) for why this is interesting along with a [subsequent addition](#) providing clarity on how to get 4662 events to fire. This detection should not be considered reliable on its own but can identify suspicious activity that warrants further investigation.

\n

Updated 01/15/2021

\n

[Spoiler](#)

\n

```
(union isfuzzy=true (SecurityEvent
| where EventID == 4662 // You need to create a SACL on the ADFS Policy Store DKM group for this event to be created.
| where ObjectServer == 'DS'
| where OperationType == 'Object Access'
//| where ObjectName contains '<GUID of ADFS Policy Store DKM Group object' This is unique to the domain. Check
description for more details.
| where ObjectType contains '5cb41ed0-0e4c-11d0-a286-00aa003049e2' // Contact Class
| where Properties contains '8d3bca50-1d7e-11d0-a081-00aa006c33ed' // Picture Attribute - Ldap-Display-Name:
thumbnailPhoto
| extend timestamp = TimeGenerated, HostCustomEntity = Computer, AccountCustomEntity = SubjectAccount),
(DeviceEvents
| where ActionType =~ \"LdapSearch\")
| where AdditionalFields.AttributeList contains \"thumbnailPhoto\"
| where AdditionalFields.DistinguishedName contains \"CN=ADFS,CN=Microsoft,CN=Program Data\" // Filter results to
show only hits related to the ADFS AD container
| extend timestamp = TimeGenerated, HostCustomEntity = DeviceName, AccountCustomEntity =
InitiatingProcessAccountName)
)
```

\n

\n

Updated 12/19/2020

\n

MSTIC has developed another [detection](#) for ADFS server key export events. This detection leverages the visibility provided by [Sysmon](#) and provides a more reliable detection method than that covered in the Windows Event Log detection. For this detection to be effective you must be collecting Sysmon Event IDs 17 and 18 into [your Azure Sentinel workspace](#).

\n

[Spoiler](#)

```
// Adjust this to use a longer timeframe to identify ADFS servers
let lookback = 6d;
// Adjust this to adjust the key export detection timeframe
let timeframe = 1d;
// Start by identifying ADFS servers to reduce FP chance
let ADFS_Servers = (
Event
| where TimeGenerated > ago(timeframe+lookback)
| where Source == \"Microsoft-Windows-Sysmon\"
| extend EventData = parse_xml(EventData).DataItem.EventData.Data
| mv-expand bagexpansion=array EventData
| evaluate bag_unpack(EventData)
| extend Key=toString(['@Name']), Value=['#text']
| evaluate pivot(Key, any(Value), TimeGenerated, Source, EventLog, Computer, EventLevel, EventLevelName, EventID,
UserName, RenderedDescription, MG, ManagementGroupName, Type, _ResourceId)
| extend process = split(Image, '\\\\', -1)[-1]
```

```
| where process =~ \"Microsoft.IdentityServer.ServiceHost.exe\"
| summarize by Computer);
// Look for ADFS servers where Named Pipes event are present
Event
| where TimeGenerated > ago(timeframe)
| where Source == \"Microsoft-Windows-Sysmon\"
| where Computer in~ (ADFS_Servers)
| extend RenderedDescription = tostring(split(RenderedDescription, '\\:')[0])
| extend EventData = parse_xml(EventData).DataItem.EventData.Data
| mv-expand bagexpansion=array EventData
| evaluate bag_unpack(EventData)
| extend Key=tostring(['@Name']), Value=['#text']
| evaluate pivot(Key, any(Value), TimeGenerated, Source, EventLog, Computer, EventLevel, EventLevelName, EventID,
UserName, RenderedDescription, MG, ManagementGroupName, Type, _ResourceId)
| extend RuleName = column_ifexists(\"RuleName\", \"\"), TechniqueId = column_ifexists(\"TechniqueId\", \"\"),
TechniqueName = column_ifexists(\"TechniqueName\", \"\")
| parse RuleName with * 'technique_id=' TechniqueId ';' * 'technique_name=' TechniqueName
| where EventID in (17,18)
// Look for Pipe related to querying the WID
| where PipeName == \"\\\\\\MICROSOFT##WID\\\\\\tsql\\\\\\query\"
| extend process = split(Image, '\\\\', -1)[-1]
// Exclude expected processes
| where process !in (\"Microsoft.IdentityServer.ServiceHost.exe\", \"Microsoft.Identity.Health.Adfs.PshSurrogate.exe\",
\"AzureADConnect.exe\", \"Microsoft.Tri.Sensor.exe\", \"wsmprovhost.exe\", \"mmc.exe\", \"sqlservr.exe\")
| extend Operation = RenderedDescription
| project-reorder TimeGenerated, EventType, Operation, process, Image, Computer, UserName
| extend HostCustomEntity = Computer, AccountCustomEntity = UserName
```

\\n

Outside of directly looking for tools, this adversary may have used custom tooling so looking for anomalous process executions or anomalous accounts logging on to our ADFS server can give us some clue when such attacks happen. Azure Sentinel provides queries that can help to:

\\n

- Find [rare anomalous process in your environment](#).
- Also look for [rare processes run by service accounts](#).
- Or uncommon processes that are in the [bottom 5% of all the process](#).
- In some instances, there is a rare command line syntax related to DLL loading, you can adjust these queries to also look at rarity on the command line.

\\n

Every environment is different and some of these queries being generic could be noisy. So, in the first step a good approach would be to limit this kind of hunting to our ADFS server.

\\n\\n\\n

Having gained a significant foothold in the on prem environment, the actor also targeted the Azure AD of some of the compromised organizations and made modifications to Azure AD settings to facilitate long term access. Microsoft has shared many relevant queries through the [Azure Sentinel GitHub](#) to identify these actions.

\\n

Updated 01/15/2021

\\n

One such activity is related to modifying domain federation trust settings. A federation trust signifies the establishment of authentication and authorization trust between two organizations so that users located in partner organizations can send authentication and authorization requests successfully.

\n

\n

- While not specifically seen in this attack, tracking federation trust modifications is important. The Azure Sentinel query for [domain federation trust settings modification](#) will alert when a user or application modifies the federation settings on the domain particularly when a new Active Directory Federated Service (ADFS) Trusted Realm object, such as a signing certificate, is added to the domain or there is an update to domain authentication from **managed** to **federated**. Modification to domain federation settings should be rare and this should be treated as a high-fidelity alert that Azure AD and Azure Sentinel users should follow up on.

\n

\n

Updated 01/15/2021

\n

The original purpose of the [STSRefreshTokenModification](#) low severity, hunting-only query was to demonstrate an event that has token validity time periods in it and demonstrate how one could monitor for anomalous/edited tokens. We have determined this event will only fire on the manual expiration of the StsRefreshToken by an admin (or the user). These types of events are most often generated when legitimate administrators troubleshoot frequent AAD (Azure AD) user sign-ins. To avoid any confusion with Solorigate investigation and hunting, we have removed this section from the blog.

\n\n

Another such activity is adding access to the Service Principal or Application. If a threat actor obtains access to an Application Administrator account, they may configure alternate authentication mechanisms for direct access to any of the scopes and services available to the Service Principal. With these privileges, the actor can add alternative authentication material for direct access to resources using this credential.

\n

\n

- Identify where the verify KeyCredential has **been updated** with [New access credential added to Application or Service Principal](#).

\n

\n

Updated 12/20/2020

\n

\n

- Identify where the verify KeyCredential **was not present** and has now has had its [First access credential added to Application or Service Principal where no credential was present](#).

\n

\n\n

[Spoiler](#)

\n

New access credential added to Application or Service Principal

```
let auditLookback = 1h;
AuditLogs
| where TimeGenerated > ago(auditLookback)
| where OperationName has_any ("Add service principal", "Certificates and secrets management") // captures "Add service principal", "Add service principal credentials", and "Update application – Certificates and secrets management" events
| where Result =~ "success"
| mv-expand target = TargetResources
| where tostring(InitiatedBy.user.userPrincipalName) has "@" or tostring(InitiatedBy.app.displayName) has "@"
| extend targetDisplayName = tostring(TargetResources[0].displayName)
| extend targetId = tostring(TargetResources[0].id)
| extend targetType = tostring(TargetResources[0].type)
| extend keyEvents = TargetResources[0].modifiedProperties
| mv-expand keyEvents
| where keyEvents.displayName =~ "KeyDescription"
```

```
| extend new_value_set = parse_json(tostring(keyEvents.newValue))
| extend old_value_set = parse_json(tostring(keyEvents.oldValue))
| where old_value_set != \"[]\"
| extend diff = set_difference(new_value_set, old_value_set)
| where isnotempty(diff)
| parse diff with * \"KeyIdentifier=\" keyIdentifier:string \",KeyType=\" keyType:string \",KeyUsage=\" keyUsage:string
\",DisplayName=\" keyDisplayName:string \"]\" *
| where keyUsage == \"Verify\" or keyUsage == \"\"
| extend UserAgent = iff(AdditionalDetails[0].key == \"User-Agent\",tostring(AdditionalDetails[0].value),\"\")
| extend InitiatingUserOrApp =
iff(isnotempty(InitiatedBy.user.userPrincipalName),tostring(InitiatedBy.user.userPrincipalName),
tostring(InitiatedBy.app.displayName))
| extend InitiatingIpAddress = iff(isnotempty(InitiatedBy.user.ipAddress), tostring(InitiatedBy.user.ipAddress),
tostring(InitiatedBy.app.ipAddress))
// The below line is currently commented out but Azure Sentinel users can modify this query to show only Application or
only Service Principal events in their environment
// where targetType =~ \"Application\" // or targetType =~ \"ServicePrincipal\"
| project-away diff, new_value_set, old_value_set
| project-reorder TimeGenerated, OperationName, InitiatingUserOrApp, InitiatingIpAddress, UserAgent,
targetDisplayName, targetId, targetType, keyDisplayName, keyType, keyUsage, keyIdentifier, CorrelationId, TenantId
| extend timestamp = TimeGenerated, AccountCustomEntity = InitiatingUserOrApp, IPCustomEntity = InitiatingIpAddress

\n
```

First access credential added to Application or Service Principal where no credential was present

```
let auditLookback = 1h;
AuditLogs
| where TimeGenerated > ago(auditLookback)
| where OperationName has_any (\"Add service principal\", \"Certificates and secrets management\") // captures \"Add
service principal\", \"Add service principal credentials\", and \"Update application – Certificates and secrets management\"
events
| where Result =~ \"success\"
| mv-expand target = TargetResources
| where tostring(InitiatedBy.user.userPrincipalName) has \"@\" or tostring(InitiatedBy.app.displayName) has \"@\"
| extend targetDisplayName = tostring(TargetResources[0].displayName)
| extend targetId = tostring(TargetResources[0].id)
| extend targetType = tostring(TargetResources[0].type)
| extend keyEvents = TargetResources[0].modifiedProperties
| mv-expand keyEvents
| where keyEvents.displayName =~ \"KeyDescription\"
| extend new_value_set = parse_json(tostring(keyEvents.newValue))
| extend old_value_set = parse_json(tostring(keyEvents.oldValue))
| where old_value_set == \"[]\"
| parse new_value_set with * \"KeyIdentifier=\" keyIdentifier:string \",KeyType=\" keyType:string \",KeyUsage=\"
keyUsage:string \",DisplayName=\" keyDisplayName:string \"]\" *
| where keyUsage == \"Verify\" or keyUsage == \"\"
| extend UserAgent = iff(AdditionalDetails[0].key == \"User-Agent\",tostring(AdditionalDetails[0].value),\"\")
| extend InitiatingUserOrApp =
iff(isnotempty(InitiatedBy.user.userPrincipalName),tostring(InitiatedBy.user.userPrincipalName),
tostring(InitiatedBy.app.displayName))
| extend InitiatingIpAddress = iff(isnotempty(InitiatedBy.user.ipAddress), tostring(InitiatedBy.user.ipAddress),
tostring(InitiatedBy.app.ipAddress))
// The below line is currently commented out but Azure Sentinel users can modify this query to show only Application or
only Service Principal events in their environment
// where targetType =~ \"Application\" // or targetType =~ \"ServicePrincipal\"
| project-away new_value_set, old_value_set
| project-reorder TimeGenerated, OperationName, InitiatingUserOrApp, InitiatingIpAddress, UserAgent,
targetDisplayName, targetId, targetType, keyDisplayName, keyType, keyUsage, keyIdentifier, CorrelationId, TenantId
| extend timestamp = TimeGenerated, AccountCustomEntity = InitiatingUserOrApp, IPCustomEntity = InitiatingIpAddress

\n
\n\n
```

This threat actor has been observed using applications to read users mailboxes within a target environment. To help identify this activity MSTIC has created a [hunting query](#) that looks for applications that have been granted mailbox read permissions followed by consent to this application. Whilst this may uncover legitimate applications hunters should validate applications granted mail read permissions genuinely require them.

\n

[Spoiler](#)

```
AuditLogs
| where Category =~ "ApplicationManagement"
| where ActivityDisplayName =~ "Add delegated permission grant"
| where Result =~ "success"
| where tostring(InitiatedBy.user.userPrincipalName) has "@" or tostring(InitiatedBy.app.displayName) has "@"
| extend props = parse_json(tostring(TargetResources[0].modifiedProperties))
| mv-expand props
| extend UserAgent = tostring(AdditionalDetails[0].value)
| extend InitiatingUser = tostring(parse_json(tostring(InitiatedBy.user)).userPrincipalName)
| extend UserIPAddress = tostring(parse_json(tostring(InitiatedBy.user)).ipAddress)
| extend DisplayName = tostring(props.displayName)
| extend Permissions = tostring(parse_json(tostring(props.newValue)))
| where Permissions has_any ("Mail.Read", "Mail.ReadWrite")
| extend PermissionsAddedTo = tostring(TargetResources[0].displayName)
| extend Type = tostring(TargetResources[0].type)
| project-away props
| join kind=leftouter(
AuditLogs
| where ActivityDisplayName has "Consent to application"
| extend AppName = tostring(TargetResources[0].displayName)
| extend AppId = tostring(TargetResources[0].id)
| project AppName, AppId, CorrelationId) on CorrelationId
| project-reorder TimeGenerated, OperationName, InitiatingUser, UserIPAddress, UserAgent, PermissionsAddedTo,
Permissions, AppName, AppId, CorrelationId
| extend timestamp = TimeGenerated, AccountCustomEntity = InitiatingUser, IPCustomEntity = UserIPAddress
```

\n

It's also advised to hunt for application consents for unexpected applications, particularly where they provide offline access to data or other high value access;

\n

- [Suspicious application consent similar to O365 Attack Toolkit](#)
- [Suspicious application consent for offline access](#)

\n\n

In addition to Azure AD pre-compromise logon hunting it is also possible to monitor for logons attempting to use invalid key material. This can help identify attempted logons using stolen key material made after key material has been rotated. This can be done by querying SigninLogs in Azure Sentinel where the ResultType is 5000811. **Please note** that if you roll your token signing certificate, there will be expected activity when searching on the above.

\n\n\n

Updated 12/27/2020

\n

The adversary will often attempt to access on-prem systems to gain further insight and mapping of the environment. As described in the **Resulting hands-on-keyboard attack** section of the [Analyzing Solorigate blog by Microsoft](#), attackers renamed windows administrative tools like adfind.exe which were then used for domain enumeration. An example of the process execution command like can look like this:

\n\n\n\n\n\n\n\n

```
C:\Windows\system32\cmd.exe /C csrss.exe -h breached.contoso.com -f (name="Domain Admins") member -list |
csrss.exe -h breached.contoso.com -f objectcategory=* > .\Mod\mod1.log
```

\n

We have provided a query in the [Azure Sentinel Github](#) which will help in detecting the [command line patterns related to ADFind usage](#). You can customize this query to look at your specific DC/ADFS servers.

\n

[Spoiler](#)

```
let startdate = 1d;
let lookupwindow = 2m;
let threshold = 3; //number of commandlines in the set below
let DCADFSServersList = dynamic (["DCServer01", "DCServer02", "ADFSserver01"]); // Enter a reference list of
hostnames for your DC/ADFS servers
let tokens =
dynamic(["objectcategory", "domainlist", "dcmodes", "adinfo", "trustdmp", "computers_pwdnotreqd", "Domain
Admins", "objectcategory=person", "objectcategory=computer", "objectcategory=*"]);
SecurityEvent
// where Computer in (DCADFSServersList) // Uncomment to limit it to your DC/ADFS servers list if specified above or
any pattern in hostnames (startswith, matches regex, etc).
| where TimeGenerated between (ago(startdate) .. now())
| where EventID == 4688
| where CommandLine has_any (tokens)
| where CommandLine matches regex "(.*)>(.*)"
| summarize Commandlines = make_set(CommandLine), LastObserved=max(TimeGenerated) by bin(TimeGenerated,
lookupwindow), Account, Computer, ParentProcessName, NewProcessName
| extend Count = array_length(Commandlines)
| where Count > threshold
```

\n

On the remote execution side, there is a pattern that can be identified related to using alternate credentials than the currently logged on user, such as when using the RUN AS feature on a Windows system and passing in explicit credentials. We have released a query that will identify when execution is occurring via [multiple explicit credentials](#) against remote targets. This requires that [Windows Event 4648](#) is being collected as part of Azure Sentinel.

\n\n

[Spoiler](#)

```
let WellKnownLocalSIDs = "S-1-5-[0-9][0-9]";
let protocols = dynamic(['cifs', 'ldap', 'RPCSS', 'host', 'HTTP', 'RestrictedKrbHost', 'TERMSRV', 'msomsdksvc', 'mssqlsvc']);
SecurityEvent
| where TimeGenerated >= ago(1d)
| where EventID == 4648
| where SubjectUserSid != 'S-1-0-0' // this is the Nobody SID which really means No security principal was included.
| where not(SubjectUserSid matches regex WellKnownLocalSIDs) //excluding system account/service account as this is
generally normal
| where TargetInfo has '/' //looking for only items that indicate an interesting protocol is included
| where Computer !has tostring(split(TargetServerName, '$')[0])
| where TargetAccount !~ tostring(split(SubjectAccount, '$')[0])
| extend TargetInfoProtocol = tolower(split(TargetInfo, '/')[0]), TargetInfoMachine = toupper(split(TargetInfo, '/')[1])
| extend TargetAccount = tolower(TargetAccount), SubjectAccount = tolower(SubjectAccount)
| extend UncommonProtocol = case(not(TargetInfoProtocol has_any (protocols)), TargetInfoProtocol, 'NotApplicable')
| summarize StartTime = min(TimeGenerated), EndTime = max(TimeGenerated), AccountsUsedCount =
dcount(TargetAccount), AccountsUsed = make_set(TargetAccount), TargetMachineCount = dcount(TargetInfoMachine),
TargetMachines = make_set(TargetInfoMachine), TargetProtocols = dcount(TargetInfoProtocol), Protocols =
make_set(TargetInfoProtocol), Processes = make_set(Process) by Computer, SubjectAccount, UncommonProtocol
| where TargetMachineCount > 1 or UncommonProtocol != 'NotApplicable'
| extend ProtocolCount = array_length(Protocols)
| extend ProtocolScore = case(
Protocols has 'rpcss' and Protocols has 'host' and Protocols has 'cifs', 10, //observed in Solorigate and depending on which
are used together the higher the score
```

```
Protocols has 'rpcss' and Protocols has 'host', 5,
Protocols has 'rpcss' and Protocols has 'cifs', 5,
Protocols has 'host' and Protocols has 'cifs', 5,
Protocols has 'ldap' or Protocols has 'rpcss' or Protocols has 'host' or Protocols has 'cifs', 1, //ldap is more commonly seen in
general, this was also seen with Solorigate but not usually to the same machines as the others above
UncommonProtocol != 'NotApplicable', 3,
0 //other protocols may be of interest, but in relation to observations for enumeration/execution in Solorigate they receive 0
)
| extend Score = ProtocolScore + ProtocolCount + AccountsUsedCount
| where Score >= 9 or (UncommonProtocol != 'NotApplicable' and Score >= 4) // Score must be 9 or better as this will
include 5 points for atleast 2 of the interesting protocols + the count of protocols (min 2) + the number of accounts used for
execution (min 2) = min of 9 OR score must be 4 or greater for an uncommon protocol
| extend TimePeriod = EndTime - StartTime //This identifies the time between start and finish for the use of the explicit
credentials, shorter time period may indicate scripted executions
| project-away UncommonProtocol
| extend timestamp = StartTime, AccountCustomEntity = SubjectAccount, HostCustomEntity = Computer
| order by Score desc
```

\n\n\n

Accessing confidential data is one of the primary motives of this attack. Data access for the attacker here relied on leveraging minted SAML tokens to access user files/email stored in the cloud via compromised AppIds. One way to detect this is when a user or application signs in using [Azure Active Directory PowerShell to access non-Active Directory resources](#).

\n\n

Microsoft Graph is one way that the attacker may be seen accessing resources and can help find what the attacker may have accessed using the Service principal Azure Active Directory sign-in logs. If you have data in your Log analytics you could easily plot a chart to see what anomalous activity is happening in your environment that is leveraging the graph.

\n

Updated 12/17/2020

\n

Note that this data type in Azure Sentinel below is only available when additional Diagnostic Logging is enabled on the workspace. Please see the instructions in the expandable section below.

\n

[Spoiler](#)

The AADServicePrincipalSignInLogs datatype will not be available in Azure Sentinel unless it is configured under Diagnostic Settings. Please see screenshots below the query.

```
AADServicePrincipalSignInLogs
| where TimeGenerated > ago(90d)
| where ResourceDisplayName == "Microsoft Graph"
| where ServicePrincipalId == "524c43c4-c484-4f7a-bd44-89d4a0d8aeab"
| summarize count() by bin(TimeGenerated, 1h)
| render timechart
```

To enable Service Principal Signin Logging, do the following:

\n

\n\n

\n\n

\n

\n

Updated 12/21/2020

\n

Additionally, below is a sample query that brings out some of the logons to Azure AD where multi factor authentication was satisfied by token based logons versus MFA via phone auth or the like. It is possible this could produce many results, so additional tuning is suggested for your environment.

\n

[Spoiler](#)

SigninLogs

```
| where TimeGenerated > ago(30d)
| where ResultType == 0
| extend additionalDetails = tostring(Status.additionalDetails)
| summarize make_set(additionalDetails), min(TimeGenerated), max(TimeGenerated) by IPAddress, UserPrincipalName
| where array_length(set_additionalDetails) == 2
| where (set_additionalDetails[1] == "MFA requirement satisfied by claim in the token\" and set_additionalDetails[0] == \"MFA requirement satisfied by claim provided by external provider\") or (set_additionalDetails[0] == \"MFA requirement satisfied by claim in the token\" and set_additionalDetails[1] == \"MFA requirement satisfied by claim provided by external provider\")
// project IPAddress, UserPrincipalName, min_TimeGenerated, max_TimeGenerated
```

\n

UPDATED 12/17/2020

\n

This attack also used Virtual Private Servers (VPS) hosts to access victim networks and can be used in conjunction with the query above. Both MSTIC and FireEye have reported attacker logon events coming from network ranges associated with VPS providers. In order to highlight these logons, MSTIC has created a new hunting query - [Signins From VPS Providers - that looks for successful signins from network ranges associated with VPS providers. This is joined with the above query, the new query looks for IPs that only display sign-ins based on tokens and not other MFA options, although this could be removed if wanted. The list of VPS ranges in the query is not comprehensive and there is significant potential for false positives so results should be investigated before responding, however it can provide very effective signal. Combining the query below with data that list VPS server ranges will make this a high-confidence hunting query.](#)

\n\n

In relation to the VPS servers section above, the previously mentioned workbook has a section that shows successful user signins from VPS (Virtual Private Server) providers where only tokens were used to authenticate. This uses the new KQL operator [ipv4_lookup](#) to evaluate if a login came from a known VPS provider network range. This operator can alternatively be used to look for all logons not coming from known ranges should your environment have a common logon source.

\n\n\n

Updated 12/20/2020

\n

Email data has been observed as a target for the Solorigate attackers, one way to monitor for potential suspicious access is to look for anomalous MailItemsAccessed volumes. MSTIC has created a specific hunting query to identify [Anomalous User Accessing Other Users Mailbox](#) which can help to identify malicious activity related to this attack. Additionally, MSTIC previously created a more generic detection - [Exchange workflow MailItemsAccessed operation anomaly](#) - which looks for time series based anomalies in MailItemsAccessed events in the OfficeActivity log.

\n

[Spoiler](#)

Anomalous access to other user's mailboxes

```
let timeframe = 14d;
let user_threshold = 1;
let folder_threshold = 5;
OfficeActivity
| where TimeGenerated > ago(timeframe)
| where Operation =~ "MailItemsAccessed"
| where ResultStatus =~ "Succeeded"
| mv-expand parse_json(Folders)
| extend folders = tostring(Folders.Path)
| where tolower(MailboxOwnerUPN) != tolower(UserId)
| extend ClientIP = if(Client_IPAddress startswith "\[", extract("^\[\\\\[\\\\\\\\]*)", 1, Client_IPAddress), Client_IPAddress)
```

```
| summarize make_set(folders), make_set(ClientInfoString), make_set(ClientIP), make_set(MailboxGuid),
make_set(MailboxOwnerUPN) by UserId
| extend folder_count = array_length(set_folders)
| extend user_count = array_length(set_MailboxGuid)
| where user_count > user_threshold or folder_count > folder_threshold
| sort by user_count desc
| project-reorder UserId, user_count, folder_count, set_MailboxOwnerUPN, set_ClientIP, set_ClientInfoString, set_folder
```

Exchange workflow MailItemsAccessed operation anomaly

```
let starttime = 14d;
let endtime = 1d;
let timeframe = 1h;
let scorethreshold = 1.5;
let percentthreshold = 50;
// Preparing the time series data aggregated hourly count of MailItemsAccessed Operation in the form of multi-value array to
use with time series anomaly function.
let TimeSeriesData =
OfficeActivity
| where TimeGenerated between (startofday(ago(starttime))).startofday(ago(endtime)))
| where OfficeWorkload=~"Exchange" and Operation =~"MailItemsAccessed" and ResultStatus =~"Succeeded"
| project TimeGenerated, Operation, MailboxOwnerUPN
| make-series Total=count() on TimeGenerated from startofday(ago(starttime)) to startofday(ago(endtime)) step timeframe;
let TimeSeriesAlerts = TimeSeriesData
| extend (anomalies, score, baseline) = series_decompose_anomalies(Total, scorethreshold, -1, 'linefit')
| mv-expand Total to typeof(double), TimeGenerated to typeof(datetime), anomalies to typeof(double), score to
typeof(double), baseline to typeof(long)
| where anomalies > 0
| project TimeGenerated, Total, baseline, anomalies, score;
// Joining the flagged outlier from the previous step with the original dataset to present contextual information
// during the anomalyhour to analysts to conduct investigation or informed decisions.
TimeSeriesAlerts | where TimeGenerated > ago(2d)
// Join against base logs since specified timeframe to retrieve records associated with the hour of anomaly
| join (
OfficeActivity
| where TimeGenerated > ago(2d)
| where OfficeWorkload=~"Exchange" and Operation =~"MailItemsAccessed" and ResultStatus =~"Succeeded"
) on TimeGenerated
```

\n

Updated 12/19/2020

\n

Targeting of email data has also been observed by other industry members including [Volexity](#) who reported attackers using PowerShell commands to export on premise Exchange mailboxes and then hosting those files on OWA servers in order to exfiltrate them.

\n

MSTIC has created detections to identify this activity at both the [OWA server](#) and [attacking host level](#) through IIS logs, and PowerShell command line logging.

\n\n

OWA exfiltration:

\n

[Spoiler](#)

```
let excludeIps = dynamic(["127.0.0.1", "::1"]);
let scriptingExt = dynamic([".aspx", ".ashx", ".asp"]);
W3CIISLog
| where csUriStem contains "/owa/"
//The actor pulls a file back but won't send it any URI params
| where isempty(csUriQuery)
| extend file_ext = tostring(split(csUriStem, "\.")[1])
```

```
//Giving your file a known scripting extension will throw an error
//rather than just serving the file as it will try to interpret the script
| where file_ext !in~ (scriptingExt)
//The actor was seen using image files, but we go wider in case they change this behaviour
//| where file_ext in~ (\"jpg\", \"jpeg\", \"png\", \"bmp\")
| extend file_name = tostring(split(csUriStem, \"^\")[1])
| where file_name != \"\"
| where cIP !in~ (excludeIps)
| project file_ext, csUriStem, file_name, Computer, cIP, sIP, TenantId, TimeGenerated
| summarize dcount(cIP), AccessingIPs=make_set(cIP), AccessTimes=make_set(TimeGenerated), Access=count() by
TenantId, file_name, Computer, csUriStem
//Collection of the exfiltration will occur only once, lets check for 2 accesses in case they mess up
//Tailor this for hunting
| where Access <= 2 and dcount_cIP == 1

\n

Host creating then removing mailbox export requests using PowerShell cmdlets:

\n

Spoiler

\n

// Adjust the timeframe to change the window events need to occur within to alert

\n

let timeframe = 1h;

\n

SecurityEvent

\n

| where Process in (\"powershell.exe\", \"cmd.exe\")

\n

| where CommandLine contains 'New-MailboxExportRequest'

\n

| summarize by Computer, timekey = bin(TimeGenerated, timeframe), CommandLine, SubjectUserName

\n

| join kind=inner (SecurityEvent

\n

| where Process in (\"powershell.exe\", \"cmd.exe\")

\n

| where CommandLine contains 'Remove-MailboxExportRequest'

\n

| summarize by Computer, timekey = bin(TimeGenerated, timeframe), CommandLine, SubjectUserName) on Computer,
timekey, SubjectUserName

\n

| extend commands = pack_array(CommandLine1, CommandLine)

\n

| summarize by timekey, Computer, tostring(commands), SubjectUserName

\n

| project-reorder timekey, Computer, SubjectUserName, ['commands']
```

\n

```
| extend HostCustomEntity = Computer, AccountCustomEntity = SubjectUserName
```

\n

\n

Updated 12/28/2020

\n

Email Delegation and later delegate access is another tactic that has been observed to gain access to user's mailboxes. We have a previously created a method to discover [Non-owner mailbox login activity](#), that can be applied here to help identify when delegates are inappropriately access email.

\n\n

[Spoiler](#)

```
let timeframe = 1d;
OfficeActivity
| where TimeGenerated >= ago(timeframe)
| where Operation == \"MailboxLogin\" and Logon_Type != \"Owner\"
| summarize count(), min(TimeGenerated), max(TimeGenerated) by Operation, OrganizationName, UserType, UserId,
MailboxOwnerUPN, Logon_Type
| extend timestamp = min_TimeGenerated, AccountCustomEntity = UserId
```

\n\n\n

Updated 12/17/2020

\n\n

MSTIC has collated network based IoCs from [MSTIC](#), [FireEye](#) and [Volexity](#) to create a network based IoC detection - [Solorigate Network Beacon](#) - that leverage multiple network focused data sources within Azure Sentinel.

\n

[Spoiler](#)

```
let domains =
dynamic([\"incomeupdate.com\", \"zupertech.com\", \"databasegalore.com\", \"panhardware.com\", \"avsmcloud.com\", \"digitalcollege.org\", \"freescanoni
let timeframe = 6h;
(union isfuzzy=true
(CommonSecurityLog
| where TimeGenerated >= ago(timeframe)
| parse Message with * (' DNSName ') *
| where DNSName in~ (domains) or DestinationHostName has_any (domains) or RequestURL has_any(domains)
| extend AccountCustomEntity = SourceUserID, HostCustomEntity = DeviceName, IPCustomEntity = SourceIP
),
(DnsEvents
| where TimeGenerated >= ago(timeframe)
| extend DNSName = Name
| where isnotempty(DNSName)
| where DNSName in~ (domains)
| extend IPCustomEntity = ClientIP
),
(VMConnection
| where TimeGenerated >= ago(timeframe)
| parse RemoteDnsCanonicalNames with * '[' DNSName '\\'] *
| where isnotempty(DNSName)
| where DNSName in~ (domains)
| extend IPCustomEntity = RemoteIp
),
(DeviceNetworkEvents
| where TimeGenerated >= ago(timeframe)
| where isnotempty(RemoteUrl)
| where RemoteUrl has_any (domains)
| extend DNSName = RemoteUrl
```

```
| extend IPCustomEntity = RemoteIP
| extend HostCustomEntity = DeviceName
)
)
\n\n
```

The avsvmcloud[.]com has been observed by several organizations as making DGA like subdomain queries as part of C2 activities. MSTIC have generated a hunting query - [Solorigate DNS Pattern](#) - to look for similar patterns of activity from other domains that might help identify other potential C2 sources.

\n

[Spoiler](#)

```
let cloudApiTerms = dynamic(["api", "east", "west"]);
DnsEvents
| where IPAddresses != "" and IPAddresses != "127.0.0.1"
| where Name endswith ".com" or Name endswith ".org" or Name endswith ".net"
| extend domain_split = split(Name, ".")
| where tostring(domain_split[-5]) != "" and tostring(domain_split[-6]) == ""
| extend sub_domain = tostring(domain_split[0])
| where sub_domain !contains "-"
| extend sub_directories = strcat(domain_split[-3], "\", domain_split[-4])
| where sub_directories has_any(cloudApiTerms)
//Based on sample communications the subdomain is always between 20 and 30 bytes
| where strlen(domain_split) < 32 or strlen(domain_split) > 20
| extend domain = strcat(tostring(domain_split[-2]), ".", tostring(domain_split[-1]))
| extend subdomain_no = countof(sub_domain, @"\d", "regex")
| extend subdomain_ch = countof(sub_domain, @"([a-z])", "regex")
| where subdomain_no > 1
| extend percentage_numerical = toreal(subdomain_no) / toreal(strlen(sub_domain)) * 100
| where percentage_numerical < 50 and percentage_numerical > 5
| summarize count(), make_set(Name), FirstSeen=min(TimeGenerated), LastSeen=max(TimeGenerated) by Name
| order by count_ asc
```

\n\n

In addition we have another query - [Solorigate Encoded Domain in URI](#)- that takes the encoding pattern the DGA uses, encodes the domains seen in signin logs and then looks for those patterns in DNS logs. This can help identify other C2 domains using the same encoding scheme.

\n

[Spoiler](#)

```
let dictionary =
dynamic(["r", "q", "3", "g", "s", "a", "l", "6", "u", "1", "i", "y", "f", "z", "o", "p", "5", "7", "2", "d", "4", "9", "b", "n", "x", "8", "c"])
let regex_bad_domains = SigninLogs
//Collect domains from tenant from from signin logs
| where TimeGenerated > ago(1d)
| extend domain = tostring(split(UserPrincipalName, "@")[0])
| where domain != ""
| summarize by domain
| extend split_domain = split(domain, ".")
//This cuts back on domains such as na.contoso.com by electing not to match on the "na" portion
| extend target_string = iff(strlen(split_domain[0]) <= 2, split_domain[1], split_domain[0])
| extend target_string = split(target_string, "-")
| mv-expand target_string
//Rip all of the alphanumeric out of the domain name
| extend string_chars = extract_all(@"([a-z0-9])", tostring(target_string))
//Guid for tracking our data
| extend guid = new_guid()
//Expand to get all of the individual chars from the domain
| mv-expand string_chars
| extend chars = tostring(string_chars)
//Conduct computation to encode the domain as per actor spec
```

```
| extend computed_char = array_index_of(dictionary, chars)
| extend computed_char = dictionary[(computed_char + 4) % array_length(dictionary)]
| summarize make_list(computed_char) by guid, domain
| extend target_encoded = tostring(strcat_array(list_computed_char, "\""))
//These are probably too small, but can be edited (expect FP's when going too small)
| where strlen(target_encoded) > 5
| distinct target_encoded
| summarize make_set(target_encoded)
//Key to join to DNS
| extend key = 1;
DnsEvents
| where TimeGenerated > ago(1d)
| summarize by Name
| extend key = 1
//For each DNS query join the malicious domain list
| join kind=inner (
regex_bad_domains
) on key
| project-away key
//Expand each malicious key for each DNS query observed
| mv-expand set_target_encoded
//IndexOf allows us to fuzzy match on the substring
| extend match = indexof(Name, set_target_encoded)
| where match > -1
```

\n\n

Updated 01/19/2021

\n

There has been additional indication that security services are being tampered with to hinder detection and investigation. While this is a common tactic, we felt that we should include this reference. The query is currently written specifically for [Potential Microsoft security services tampering](#), but can easily be adapted to identify other security services.

\n

Spoiler

```
let includeProc = dynamic(["sc.exe","net1.exe","net.exe","taskkill.exe","cmd.exe","powershell.exe"]);
let action = dynamic(["stop","disable","delete"]);
let service1 = dynamic(["sense","windefend","mssecflt"]);
let service2 = dynamic(["sense","windefend","mssecflt","healthservice"]);
let params1 = dynamic(["-DisableRealtimeMonitoring","-DisableBehaviorMonitoring","-DisableIOAVProtection"]);
let params2 = dynamic(["sgrmbroker.exe","mssense.exe"]);
let regparams1 = dynamic(["reg add \"HKLM\\SOFTWARE\\Policies\\Microsoft\\Windows Defender\"", "reg add \"HKLM\\SOFTWARE\\Policies\\Microsoft\\Windows Advanced Threat Protection\""]);
let regparams2 = dynamic(["ForceDefenderPassiveMode", "DisableAntiSpyware"]);
let regparams3 = dynamic(["sense", "windefend"]);
let regparams4 = dynamic(["demand", "disabled"]);
let regparams5 = dynamic(["reg add \"HKLM\\SYSTEM\\CurrentControlSet\\services\\HealthService\"", "reg add \"HKLM\\SYSTEM\\CurrentControlSet\\Services\\Sense\"", "reg add \"HKLM\\SYSTEM\\CurrentControlSet\\Services\\WinDefend\"", "reg add \"HKLM\\SYSTEM\\CurrentControlSet\\Services\\MsSecFlt\"", "reg add \"HKLM\\SYSTEM\\CurrentControlSet\\Services\\DiagTrack\"", "reg add \"HKLM\\SYSTEM\\CurrentControlSet\\Services\\SgrmBroker\"", "reg add \"HKLM\\SYSTEM\\CurrentControlSet\\Services\\SgrmAgent\"", "reg add \"HKLM\\SYSTEM\\CurrentControlSet\\Services\\AATPSensorUpdater\"", "reg add \"HKLM\\SYSTEM\\CurrentControlSet\\Services\\AATPSensor\"", "reg add \"HKLM\\SYSTEM\\CurrentControlSet\\Services\\mpssvc\""]);
let regparams6 = dynamic(["/d 4;/d \"4\";/d 0x00000004"]);
let regparams7 = dynamic(["/d 1;/d \"1\";/d 0x00000001"]);
let timeframe = 1d;
(union isfuzzy=true
(
SecurityEvent
```

```

| where TimeGenerated >= ago(timeframe)
| where EventID == 4688
| extend ProcessName = tostring(split(NewProcessName, "\\")[1])
| where ProcessName in~ (includeProc)
| where (CommandLine has_any (action) and CommandLine has_any (service1))
or (CommandLine has_any (params1) and CommandLine has 'Set-MpPreference' and CommandLine has '$true')
or (CommandLine has_any (params2) and CommandLine has '/IM')
or (CommandLine has_any (regparams5) and CommandLine has 'Start' and CommandLine has_any (regparams6))
or (CommandLine has_any (regparams1) and CommandLine has_any (regparams2) and CommandLine has_any
(regparams7))
or (CommandLine has 'start' and CommandLine has 'config' and CommandLine has_any (regparams3) and
CommandLine has_any (regparams4))
| project TimeGenerated, Computer, Account, AccountDomain, ProcessName, ProcessNameFullPath = NewProcessName,
EventID, Activity, CommandLine, EventSourceName, Type
),
(
Event
| where TimeGenerated >= ago(timeframe)
| where Source =~ \"Microsoft-Windows-SENSE\"
| where EventID == 87 and ParameterXml in (\"<Param>sgmbroker</Param>\", \"<Param>WinDefend</Param>\")
| project TimeGenerated, Computer, Account = UserName, EventID, Activity = RenderedDescription, EventSourceName =
Source, Type
),
(
DeviceProcessEvents
| where TimeGenerated >= ago(timeframe)
| where InitiatingProcessFileName in~ (includeProc)
| where (InitiatingProcessCommandLine has_any(action) and InitiatingProcessCommandLine has_any (service2) and
InitiatingProcessParentFileName != 'cscrip.exe')
or (InitiatingProcessCommandLine has_any (params1) and InitiatingProcessCommandLine has 'Set-MpPreference' and
InitiatingProcessCommandLine has '$true')
or (InitiatingProcessCommandLine has_any (params2) and InitiatingProcessCommandLine has '/IM')
or (InitiatingProcessCommandLine has_any (regparams5) and InitiatingProcessCommandLine has 'Start' and
InitiatingProcessCommandLine has_any (regparams6))
or (InitiatingProcessCommandLine has_any (regparams1) and InitiatingProcessCommandLine has_any (regparams2) and
InitiatingProcessCommandLine has_any (regparams7))
or (InitiatingProcessCommandLine has_any('start') and InitiatingProcessCommandLine has 'config' and
InitiatingProcessCommandLine has_any (regparams3) and InitiatingProcessCommandLine has_any (regparams4))
| extend Account = iff(isnotempty(InitiatingProcessAccountUpn), InitiatingProcessAccountUpn,
InitiatingProcessAccountName), Computer = DeviceName
| project TimeGenerated, Computer, Account, AccountDomain, ProcessName = InitiatingProcessFileName,
ProcessNameFullPath = FolderPath, Activity = ActionType, CommandLine = InitiatingProcessCommandLine, Type,
InitiatingProcessParentFileName
)
)
| extend timestamp = TimeGenerated, AccountCustomEntity = Account, HostCustomEntity = Computer

\n\n

```

In addition we have created a query in Azure Sentinel - [Solorigate Defender Detections](#) - to collate the range of Defender detections that are now deployed. This query can be used to get an overview of such alerts and the hosts they relate to.

\n

[Spoiler](#)

```

DeviceInfo
| extend DeviceName = tolower(DeviceName)
| join (SecurityAlert
| where ProviderName =~ \"MDATP\"
| extend ThreatName = tostring(parse_json(ExtendedProperties).ThreatName)
| where ThreatName has \"Solarigate\"
| extend HostCustomEntity = tolower(CompromisedEntity)
| take 10) on $left.DeviceName == $right.HostCustomEntity
| project TimeGenerated, DisplayName, ThreatName, CompromisedEntity, PublicIP, MachineGroup, AlertSeverity,

```

Description, LoggedOnUsers, DeviceId, TenantId
| extend timestamp = TimeGenerated, IPCustomEntity = PublicIP

\n\n\n

Additionally, as a cloud native SIEM Azure Sentinel can not only collect raw data from various disparate logs but it also gets alerts from various security products. For example, M365 Defender has a range of alerts for various attack components like SolarWinds malicious binaries, network traffic to the compromised domains, DNS queries for known patterns associated with SolarWinds compromise that can flow into Sentinel. Combining these alerts with other raw logs and additional data sources provides the security team with additional insights as well as a complete picture of nature and the scope of attack.

\n\n\n

Many of these queries have been incorporated into the related [hunting workbook](#).

\n

List of all Azure Sentinel Queries from each section

\n

Updated 01/15/2021

\n\n\n

[Recent Nation-State Cyber Attacks](#)

\n

[Behavior:Win32/Solorigate.C!dha threat description - Microsoft Security Intelligence](#)

\n

[Customer guidance on recent nation-state cyberattacks](#)

\n

[FireEye Advisory: Highly Evasive Attacker Leverages SolarWinds Supply Chain to Compromise Multiple Global Victims With SUNBURST Backdoor](#)

\n

[FireEye GitHub page: Sunburst Countermeasures](#)

\n

[DHS Directive](#)

\n

[SolarWinds Security Advisory](#)

\n

[FalconFriday – Fireeye Red Team Tool Countermeasures KQL Queries](#)

\n

[Microsoft-365-Defender-Hunting-Queries: Sample queries for Advanced hunting in Microsoft 365 Defender \(github.com\)](#)

\n

[Azure Sentinel SolarWinds Post Compromise Hunting Workbook](#)

\n

[Azure Sentinel SolarWinds Post Compromise Notebook](#)

\n

Updated 12/18/2020

\n

[New Threat analytics report shares the latest intelligence on recent nation-state cyber attacks - Microsoft Tech Community](#)

\n

in [Microsoft Threat Protection \(MTP\) portal](#). Note that if you do not have Microsoft Threat Protection this link will not work for you.

\n\n\n\n

To hunt for similar TTPs used in this attack, a good place to start is to build an inventory of the machines that have SolarWinds Orion components. Organizations might already have a software inventory management system to indicate hosts where the SolarWinds application is installed. Alternatively, Azure Sentinel could be leveraged to run a simple query to gather similar details. Azure Sentinel collects data from multiple different logs that could be used to gather this information. For example, through the recently released [Microsoft 365 Defender connector](#), security teams can now easily ingest Microsoft 365 raw data into Azure Sentinel. Using the ingested data, a simple query like below can be written that will pull the hosts with SolarWinds process running in last 30 days based on Process execution either via host on boarded to Sentinel or on boarded via Microsoft Defender for Endpoints (MDE). The query also leverages the Sysmon logs that a lot of customers are collecting from their environment to surface the machines that have SolarWinds running on them. Similar queries that leverage M365 raw data could also be run from the M365's Advanced hunting portal.

\n\n

[SolarWinds Inventory check query](#)

\n\n

\n

```
let timeframe = 30d;
```

\n

```
(union isfuzzy=true
```

\n

```
(
```

\n

```
SecurityEvent
```

\n

```
| where TimeGenerated >= ago(timeframe)
```

\n

```
| where EventID == '4688'
```

\n

```
| where tolower(NewProcessName) has 'solarwinds'
```

\n

```
| extend MachineName = Computer , Process = NewProcessName
```

\n

```
| summarize StartTime = min(TimeGenerated), EndTime = max(TimeGenerated), MachineCount = dcount(MachineName), AccountCount = dcount(Acco
```

\n

```
),
```

\n

```
(
```

\n

```
DeviceProcessEvents
```

\n

```
| where TimeGenerated >= ago(timeframe)
```

\n

```
| where tolower(InitiatingProcessFolderPath) has 'solarwinds'

\n

| extend MachineName = DeviceName , Process = InitiatingProcessFolderPath, Account = AccountName

\n

|

summarize StartTime = min(TimeGenerated), EndTime = max(TimeGenerated), MachineCount = dcount(MachineName), AccountCount = dcount(Acco

\n

),

\n

(

\n

Event

\n

| where TimeGenerated >= ago(timeframe)

\n

| where Source == \"Microsoft-Windows-Sysmon\"

\n

| where EventID == 1

\n

| extend Image = EventDetail.[4],[\"#text\"]

\n

| where tolower(Image) has 'solarwinds'

\n

| extend MachineName = Computer , Process = Image, Account = UserName

\n

|

summarize StartTime = min(TimeGenerated), EndTime = max(TimeGenerated), MachineCount = dcount(MachineName), AccountCount = dcount(Acco

\n

)

\n

)

\n

\n\n
```

Updated 12/30/2020

\n

On systems where the malicious SolarWinds DLL (SolarWinds.Orion.Core.BusinessLayer.dll) is running, it is known that the attacker used a hardcoded named pipe '583da945-62af-10e8-4902-a8f205c72b2e' to conduct various checks as well as to ensure only one instance of the backdoor was running. The use of named pipes by malware is not uncommon as it provides a mechanism for communication between processes. This activity by the malware can be detected if you are collecting Sysmon (Event Id 17/18) or Security Event Id 5145 in your Azure Sentinel workspace. The [Solorigate Named Pipe detection](#) should not be considered reliable on its own as the creation of just the hardcoded named pipe does not indicate that the malicious code was completely triggered, and the machine beacons out or received additional commands. However, presence of this is definitely suspicious and should warrant further in-depth investigation.

```
\nlet timeframe = 1d;
(union isfuzzy=true
(Event
| where TimeGenerated >= ago(timeframe)
| where Source == \"Microsoft-Windows-Sysmon\"
| where EventID in (17,18)
| extend EvData = parse_xml(EventData)
| extend EventDetail = EvData.DataItem.EventData.Data
| extend NamedPipe = EventDetail.[5].[\"#text\"]
| extend ProcessDetail = EventDetail.[6].[\"#text\"]
| where NamedPipe contains '583da945-62af-10e8-4902-a8f205c72b2e'
| extend Account = UserName
| project-away EventDetail, EvData
),
(
SecurityEvent
| where TimeGenerated >= ago(timeframe)
| where EventID == '5145'
| where AccessList has '%4418' // presence of CreatePipeInstance value
| where RelativeTargetName contains '583da945-62af-10e8-4902-a8f205c72b2e'
)
)
| extend timestamp = TimeGenerated, AccountCustomEntity = Account, HostCustomEntity = Computer\n
```

Privilege Escalation

\n

Once the adversary acquires an initial foothold on a system thru the SolarWinds process they will have System account level access, the attacker will then attempt to elevate to domain admin level access to the environment. The Microsoft Threat Intelligence Center (MSTIC) team has already delivered multiple queries into Azure Sentinel that identify similar TTPs and many are also available in M365. These methodologies are not specific to just this threat actor or this attack but have been seen in various attack campaigns.

\n

Identifying abnormal logon activities or additions to privileged groups is one way to identify privilege escalation.

\n

Updated 12/17/2020

\n

\n

- Checking for [hosts with new logons](#) to identify potential lateral movement by the attacker.

\n

- Look for any new [account being created and added to built-in administrators group](#).

\n

- Look for any [user account added to privileged built in domain local or global groups](#), including adding accounts to a domain privileged group such as Enterprise Admins, Cert Publishers or DnsAdmins.

\n

- Monitor for [rare activity by a high-value account](#) carried out on a system or service.

\n

\n

Related to this attack, in some environments service account credentials had been granted administrative privileges. The above queries can be modified to remove the condition of focusing “User” accounts by commenting the query to include service accounts in the scope where applicable:

\n\n

```
// where AccountType == \"User\"
```

\n\n

Please see the [Azure Sentinel Github](#) for additional queries and hunting ideas related to Accounts under the Detections and Hunting Queries sections for AuditLogs, and SecurityEvents

\n

Microsoft 365 Defender team has also shared quite a few sample queries for use in their [advanced hunting](#) portal that could be leveraged to detect this part of the attack. Additionally, the logic for many of the Azure Sentinel queries can also be transformed to equivalent queries for Microsoft 365 Defender, that could be run in their Advanced Hunting Portal.

\n

Microsoft 365 Defender has an upcoming complimentary blog that will be updated here once available.

\n\n

Certificate Export

\n

The next step in the attack was stealing the certificate that signs SAML tokens from the federation server (ADFS) called a Token Signing Cert (TSC). SAML Tokens are basically XML representations of claims. You can read more about ADFS in [What is federation with Azure AD? | Microsoft Docs](#) and SAML at [Azure Single Sign On SAML Protocol - Microsoft identity platform | Microsoft Docs](#). The process is as follows:

\n

- \n
1. A client requests a SAML token from an ADFS Server by authenticating to that server using Windows credentials.
\n
2. The ADFS server issues a SAML token to the client.
\n
3. The SAML token is signed with a certificate associated with the server.
\n
4. The client then presents the SAML token to the application that it needs access to.
\n
5. The signature over the SAML token tells the application that the security token service issued the token and grants access to the client.
\n

\n\n

ADFS Key Extraction

\n

Updated 01/15/2021

\n

The implication of stealing the Token Signing Cert (TSC) is that once the certificate has been acquired, the actor can forge SAML (Security Assertions Markup Language) tokens with whatever claims and lifetime they choose, then sign it with the certificate that has been acquired. Microsoft continues to strongly recommend securing your AD FS (Active Directory Federation Service) TSC because if these TSC's are acquired by a bad actor, this then enables the actor to forge SAML tokens that impersonate highly privileged accounts. There are publicly available pen-testing tools like [ADFSDump](#) and [ADFSpoof](#) that help with extracting required information from the AD FS configuration database to generate the forged security tokens. While we have not confirmed these specific tools were used in this attack, they are useful for simulating the attack behavior or executing a similar attack and therefore, Microsoft has created a high-fidelity detection related to this for M365 Defender:

\n

- \n
• **ADFS private key extraction** which detects ADFS private key extraction patterns from tools such as ADFSdump.
\n

\n

Note: Any M365 Defender alert can be seen in Azure Sentinel Security Alerts or in the M365 security portal.

\n

Updated 01/15/2021

\n

The TTP (tactics, techniques, and procedures) observed in the Solorigate attack is the creation of a legitimate SAML token used to authenticate as any user. One way an attacker could achieve this is by compromising AD FS key material. Microsoft has a new detection for this as stated above and for Azure Sentinel has also created a Windows Event Log based detection that indicates an [ADFS DKM Master Key Export](#). As part of the update for this query to the Azure Sentinel GitHub, there is a [detailed write up](#) for why this is interesting along with a [subsequent addition](#) providing clarity on how to get 4662 events to fire. This detection should not be considered reliable on its own but can identify suspicious activity that warrants further investigation.

\n

Updated 01/15/2021

\n\n

```
(union isfuzzy=true (SecurityEvent
| where EventID == 4662 // You need to create a SACL on the ADFS Policy Store DKM group for this event to be created.
| where ObjectServer == 'DS'
| where OperationType == 'Object Access'
//| where ObjectName contains '<GUID of ADFS Policy Store DKM Group object' This is unique to the domain. Check
description for more details.
| where ObjectType contains '5cb41ed0-0e4c-11d0-a286-00aa003049e2' // Contact Class
| where Properties contains '8d3bca50-1d7e-11d0-a081-00aa006c33ed' // Picture Attribute - Ldap-Display-Name:
thumbnailPhoto
| extend timestamp = TimeGenerated, HostCustomEntity = Computer, AccountCustomEntity = SubjectAccount),
(DeviceEvents
| where ActionType =~ \"LdapSearch\")
| where AdditionalFields.AttributeList contains \"thumbnailPhoto\"
| where AdditionalFields.DistinguishedName contains \"CN=ADFS,CN=Microsoft,CN=Program Data\" // Filter results to
show only hits related to the ADFS AD container
| extend timestamp = TimeGenerated, HostCustomEntity = DeviceName, AccountCustomEntity =
InitiatingProcessAccountName)
)
```

\n\n

Updated 12/19/2020

\n

MSTIC has developed another [detection](#) for ADFS server key export events. This detection leverages the visibility provided by [Sysmon](#) and provides a more reliable detection method than that covered in the Windows Event Log detection. For this detection to be effective you must be collecting Sysmon Event IDs 17 and 18 into [your Azure Sentinel workspace](#).

```
\n// Adjust this to use a longer timeframe to identify ADFS servers
let lookback = 6d;
// Adjust this to adjust the key export detection timeframe
let timeframe = 1d;
// Start be identifying ADFS servers to reduce FP chance
let ADFS_Servers = (
Event
| where TimeGenerated > ago(timeframe+lookback)
| where Source == \"Microsoft-Windows-Sysmon\"
| extend EventData = parse_xml(EventData).DataItem.EventData.Data
| mv-expand bagexpansion=array EventData
| evaluate bag_unpack(EventData)
| extend Key=tostring(['@Name']), Value=['#text']
| evaluate pivot(Key, any(Value), TimeGenerated, Source, EventLog, Computer, EventLevel, EventLevelName, EventID,
UserName, RenderedDescription, MG, ManagementGroupName, Type, _ResourceId)
| extend process = split(Image, '\\\\', -1)[-1]
| where process =~ \"Microsoft.IdentityServer.ServiceHost.exe\"
| summarize by Computer);
// Look for ADFS servers where Named Pipes event are present
Event
| where TimeGenerated > ago(timeframe)
```

```
| where Source == \"Microsoft-Windows-Sysmon\"
| where Computer in~ (ADFS_Servers)
| extend RenderedDescription = tostring(split(RenderedDescription, '\')[0])
| extend EventData = parse_xml(EventData).DataItem.EventData.Data
| mv-expand bagexpansion=array EventData
| evaluate bag_unpack(EventData)
| extend Key=tostring(['@Name']), Value=['#text']
| evaluate pivot(Key, any(Value), TimeGenerated, Source, EventLog, Computer, EventLevel, EventLevelName, EventID,
UserName, RenderedDescription, MG, ManagementGroupName, Type, _ResourceId)
| extend RuleName = column_ifexists(\"RuleName\", \"\"), TechniqueId = column_ifexists(\"TechniqueId\", \"\"),
TechniqueName = column_ifexists(\"TechniqueName\", \"\")
| parse RuleName with * 'technique_id=' TechniqueId ' * 'technique_name=' TechniqueName
| where EventID in (17,18)
// Look for Pipe related to querying the WID
| where PipeName == \"\\\\MICROSOFT##WID\\\\tsql\\\\query\"
| extend process = split(Image, '\\', -1)[-1]
// Exclude expected processes
| where process !in (\"Microsoft.IdentityServer.ServiceHost.exe\", \"Microsoft.Identity.Health.Adfs.PshSurrogate.exe\",
\"AzureADConnect.exe\", \"Microsoft.Tri.Sensor.exe\", \"wsmprovhost.exe\", \"mmc.exe\", \"sqlservr.exe\")
| extend Operation = RenderedDescription
| project-reorder TimeGenerated, EventType, Operation, process, Image, Computer, UserName
| extend HostCustomEntity = Computer, AccountCustomEntity = UserName
```

Outside of directly looking for tools, this adversary may have used custom tooling so looking for anomalous process executions or anomalous accounts logging on to our ADFS server can give us some clue when such attacks happen. Azure Sentinel provides queries that can help to:

- ```
\n
 \n
 • Find rare anomalous process in your environment.
 \n
 • Also look for rare processes run by service accounts
 \n
 • Or uncommon processes that are in the bottom 5% of all the process.
 \n
 • In some instances, there is a rare command line syntax related to DLL loading, you can adjust these queries to also look at rarity on the command line.
 \n
```

```
\n
Every environment is different and some of these queries being generic could be noisy. So, in the first step a good approach would be to limit this kind of hunting to our ADFS server.
```

```
\n\n

Azure Active Directory Hunting


```

```
\n
Having gained a significant foothold in the on prem environment, the actor also targeted the Azure AD of some of the compromised organizations and made modifications to Azure AD settings to facilitate long term access. Microsoft has shared many relevant queries through the Azure Sentinel GitHub to identify these actions.
```

```
\n
Updated 01/15/2021
```

```
\n
One such activity is related to modifying domain federation trust settings. A federation trust signifies the establishment of authentication and authorization trust between two organizations so that users located in partner organizations can send authentication and authorization requests successfully.
```

```
\n
 \n
```

- While not specifically seen in this attack, tracking federation trust modifications is important. The Azure Sentinel query for [domain federation trust settings modification](#) will alert when a user or application modifies the federation settings on the domain particularly when a new Active Directory Federated Service (ADFS) Trusted Realm object, such as a signing certificate, is added to the domain or there is an update to domain authentication from **managed** to **federated**. Modification to domain federation settings should be rare and this should be treated as a high-fidelity alert that Azure AD and Azure Sentinel users should follow up on.

\n

\n

Updated 01/15/2021

\n

The original purpose of the [STSRefreshTokenModification](#) low severity, hunting-only query was to demonstrate an event that has token validity time periods in it and demonstrate how one could monitor for anomalous/edited tokens. We have determined this event will only fire on the manual expiration of the StsRefreshToken by an admin (or the user). These types of events are most often generated when legitimate administrators troubleshoot frequent AAD (Azure AD) user sign-ins. To avoid any confusion with Solorigate investigation and hunting, we have removed this section from the blog.

\n\n

Another such activity is adding access to the Service Principal or Application. If a threat actor obtains access to an Application Administrator account, they may configure alternate authentication mechanisms for direct access to any of the scopes and services available to the Service Principal. With these privileges, the actor can add alternative authentication material for direct access to resources using this credential.

\n

\n

- Identify where the verify KeyCredential has **been updated** with [New access credential added to Application or Service Principal](#).

\n

\n

Updated 12/20/2020

\n

\n

- Identify where the verify KeyCredential **was not present** and has now has had its [First access credential added to Application or Service Principal where no credential was present](#).

\n

\n\n\n

#### **New access credential added to Application or Service Principal**

```
let auditLookback = 1h;
AuditLogs
| where TimeGenerated > ago(auditLookback)
| where OperationName has_any ("Add service principal", "Certificates and secrets management") // captures "Add
service principal", "Add service principal credentials", and "Update application – Certificates and secrets management"
events
| where Result =~ "success"
| mv-expand target = TargetResources
| where tostring(InitiatedBy.user.userPrincipalName) has "@" or tostring(InitiatedBy.app.displayName) has "@"
| extend targetDisplayName = tostring(TargetResources[0].displayName)
| extend targetId = tostring(TargetResources[0].id)
| extend targetType = tostring(TargetResources[0].type)
| extend keyEvents = TargetResources[0].modifiedProperties
| mv-expand keyEvents
| where keyEvents.displayName =~ "KeyDescription"
| extend new_value_set = parse_json(tostring(keyEvents.newValue))
| extend old_value_set = parse_json(tostring(keyEvents.oldValue))
| where old_value_set != "[]"
| extend diff = set_difference(new_value_set, old_value_set)
| where isnotempty(diff)
| parse diff with * "KeyIdentifier=" keyIdentifier:string "\",KeyType=" keyType:string "\",KeyUsage=" keyUsage:string
```

```

\",DisplayName=\" keyDisplayName:string \"]\" *
| where keyUsage == \"Verify\" or keyUsage == \"\"
| extend UserAgent = iff(AdditionalDetails[0].key == \"User-Agent\",tostring(AdditionalDetails[0].value),\"\")
| extend InitiatingUserOrApp =
iff(isnotempty(InitiatedBy.user.userPrincipalName),tostring(InitiatedBy.user.userPrincipalName),
tostring(InitiatedBy.app.displayName))
| extend InitiatingIpAddress = iff(isnotempty(InitiatedBy.user.ipAddress), tostring(InitiatedBy.user.ipAddress),
tostring(InitiatedBy.app.ipAddress))
// The below line is currently commented out but Azure Sentinel users can modify this query to show only Application or
only Service Principal events in their environment
//| where targetType =~ \"Application\" // or targetType =~ \"ServicePrincipal\"
| project-away diff, new_value_set, old_value_set
| project-reorder TimeGenerated, OperationName, InitiatingUserOrApp, InitiatingIpAddress, UserAgent,
targetDisplayName, targetId, targetType, keyDisplayName, keyType, keyUsage, keyIdentifier, CorrelationId, TenantId
| extend timestamp = TimeGenerated, AccountCustomEntity = InitiatingUserOrApp, IPCustomEntity = InitiatingIpAddress

\n

```

**First access credential added to Application or Service Principal where no credential was present**

```

let auditLookback = 1h;
AuditLogs
| where TimeGenerated > ago(auditLookback)
| where OperationName has_any (\"Add service principal\", \"Certificates and secrets management\") // captures \"Add
service principal\", \"Add service principal credentials\", and \"Update application – Certificates and secrets management\"
events
| where Result =~ \"success\"
| mv-expand target = TargetResources
| where tostring(InitiatedBy.user.userPrincipalName) has \"@\" or tostring(InitiatedBy.app.displayName) has \"@\"
| extend targetDisplayName = tostring(TargetResources[0].displayName)
| extend targetId = tostring(TargetResources[0].id)
| extend targetType = tostring(TargetResources[0].type)
| extend keyEvents = TargetResources[0].modifiedProperties
| mv-expand keyEvents
| where keyEvents.displayName =~ \"KeyDescription\"
| extend new_value_set = parse_json(tostring(keyEvents.newValue))
| extend old_value_set = parse_json(tostring(keyEvents.oldValue))
| where old_value_set == \"[]\"
| parse new_value_set with * \"KeyIdentifier=\" keyIdentifier:string \",KeyType=\" keyType:string \",KeyUsage=\"
keyUsage:string \",DisplayName=\" keyDisplayName:string \"]\" *
| where keyUsage == \"Verify\" or keyUsage == \"\"
| extend UserAgent = iff(AdditionalDetails[0].key == \"User-Agent\",tostring(AdditionalDetails[0].value),\"\")
| extend InitiatingUserOrApp =
iff(isnotempty(InitiatedBy.user.userPrincipalName),tostring(InitiatedBy.user.userPrincipalName),
tostring(InitiatedBy.app.displayName))
| extend InitiatingIpAddress = iff(isnotempty(InitiatedBy.user.ipAddress), tostring(InitiatedBy.user.ipAddress),
tostring(InitiatedBy.app.ipAddress))
// The below line is currently commented out but Azure Sentinel users can modify this query to show only Application or
only Service Principal events in their environment
//| where targetType =~ \"Application\" // or targetType =~ \"ServicePrincipal\"
| project-away new_value_set, old_value_set
| project-reorder TimeGenerated, OperationName, InitiatingUserOrApp, InitiatingIpAddress, UserAgent,
targetDisplayName, targetId, targetType, keyDisplayName, keyType, keyUsage, keyIdentifier, CorrelationId, TenantId
| extend timestamp = TimeGenerated, AccountCustomEntity = InitiatingUserOrApp, IPCustomEntity = InitiatingIpAddress

\n\n

```

Updated 12/19/2020

\n

This threat actor has been observed using applications to read users mailboxes within a target environment. To help identify this activity MSTIC has created a [hunting query](#) that looks for applications that have been granted mailbox read permissions followed by consent to this application. Whilst this may uncover legitimate applications hunters should validate applications granted mail read permissions genuinely require them.

```

\nAuditLogs
| where Category =~ \"ApplicationManagement\"
| where ActivityDisplayName =~ \"Add delegated permission grant\"
| where Result =~ \"success\"
| where tostring(InitiatedBy.user.userPrincipalName) has \"@\" or tostring(InitiatedBy.app.displayName) has \"@\"
| extend props = parse_json(tostring(TargetResources[0].modifiedProperties))
| mv-expand props
| extend UserAgent = tostring(AdditionalDetails[0].value)
| extend InitiatingUser = tostring(parse_json(tostring(InitiatedBy.user)).userPrincipalName)
| extend UserIPAddress = tostring(parse_json(tostring(InitiatedBy.user)).ipAddress)
| extend DisplayName = tostring(props.displayName)
| extend Permissions = tostring(parse_json(tostring(props.newValue)))
| where Permissions has_any (\"Mail.Read\", \"Mail.ReadWrite\")
| extend PermissionsAddedTo = tostring(TargetResources[0].displayName)
| extend Type = tostring(TargetResources[0].type)
| project-away props
| join kind=leftouter(
AuditLogs
| where ActivityDisplayName has \"Consent to application\"
| extend AppName = tostring(TargetResources[0].displayName)
| extend AppId = tostring(TargetResources[0].id)
| project AppName, AppId, CorrelationId) on CorrelationId
| project-reorder TimeGenerated, OperationName, InitiatingUser, UserIPAddress, UserAgent, PermissionsAddedTo,
Permissions, AppName, AppId, CorrelationId
| extend timestamp = TimeGenerated, AccountCustomEntity = InitiatingUser, IPCustomEntity = UserIPAddress\n

```

It's also advised to hunt for application consents for unexpected applications, particularly where they provide offline access to data or other high value access;

- ```

\n
\n

  - Suspicious application consent similar to O365 Attack Toolkit
\n

  - Suspicious application consent for offline access
\n

```

\n

Updated 12/17/2020 (moved location)

\n

In addition to Azure AD pre-compromise logon hunting it is also possible to monitor for logons attempting to use invalid key material. This can help identify attempted logons using stolen key material made after key material has been rotated. This can be done by querying SigninLogs in Azure Sentinel where the ResultType is 5000811. **Please note** that if you roll your token signing certificate, there will be expected activity when searching on the above.

\n\n

Recon and Remote Execution

\n

Updated 12/27/2020

\n

The adversary will often attempt to access on-prem systems to gain further insight and mapping of the environment. As described in the **Resulting hands-on-keyboard attack** section of the [Analyzing Solorigate blog by Microsoft](#), attackers renamed windows administrative tools like adfind.exe which were then used for domain enumeration. An example of the process execution command like can look like this:

\n\n\n\n\n\n\n\n

```

C:\Windows\system32\cmd.exe /C csrss.exe -h breached.contoso.com -f (name=\"Domain Admins\") member -list |
csrss.exe -h breached.contoso.com -f objectcategory=* > .\Mod\mod1.log

```

\n

We have provided a query in the [Azure Sentinel Github](#) which will help in detecting the [command line patterns related to ADFind usage](#). You can customize this query to look at your specific DC/ADFS servers.

```
\nlet startdate = 1d;
let lookupwindow = 2m;
let threshold = 3; //number of commandlines in the set below
let DCADFSServersList = dynamic ([\"DCServer01\", \"DCServer02\", \"ADFSserver01\"]); // Enter a reference list of
hostnames for your DC/ADFS servers
let tokens =
dynamic([\"objectcategory\", \"domainlist\", \"dcmodes\", \"adinfo\", \"trustdmp\", \"computers_pwdnotreqd\", \"Domain
Admins\", \"objectcategory=person\", \"objectcategory=computer\", \"objectcategory=*\"]);
SecurityEvent
// where Computer in (DCADFSServersList) // Uncomment to limit it to your DC/ADFS servers list if specified above or
any pattern in hostnames (startswith, matches regex, etc).
| where TimeGenerated between (ago(startdate) .. now())
| where EventID == 4688
| where CommandLine has_any (tokens)
| where CommandLine matches regex \"(.*)>(.*)\"
| summarize Commandlines = make_set(CommandLine), LastObserved=max(TimeGenerated) by bin(TimeGenerated,
lookupwindow), Account, Computer, ParentProcessName, NewProcessName
| extend Count = array_length(Commandlines)
| where Count > threshold\n
```

On the remote execution side, there is a pattern that can be identified related to using alternate credentials than the currently logged on user, such as when using the RUN AS feature on a Windows system and passing in explicit credentials. We have released a query that will identify when execution is occurring via [multiple explicit credentials](#) against remote targets. This requires that [Windows Event 4648](#) is being collected as part of Azure Sentinel.

```
\n\nlet WellKnownLocalSIDs = \"S-1-5-[0-9][0-9]$\";
let protocols = dynamic(['cifs', 'ldap', 'RPCSS', 'host', 'HTTP', 'RestrictedKrbHost', 'TERMSRV', 'msomsdksvc', 'mssqlsvc']);
SecurityEvent
| where TimeGenerated >= ago(1d)
| where EventID == 4648
| where SubjectUserSid != 'S-1-0-0' // this is the Nobody SID which really means No security principal was included.
| where not(SubjectUserSid matches regex WellKnownLocalSIDs) //excluding system account/service account as this is
generally normal
| where TargetInfo has '/' //looking for only items that indicate an interesting protocol is included
| where Computer !has tostring(split(TargetServerName, '$')[0])
| where TargetAccount !~ tostring(split(SubjectAccount, '$')[0])
| extend TargetInfoProtocol = tolower(split(TargetInfo, '/')[0]), TargetInfoMachine = toupper(split(TargetInfo, '/')[1])
| extend TargetAccount = tolower(TargetAccount), SubjectAccount = tolower(SubjectAccount)
| extend UncommonProtocol = case(not(TargetInfoProtocol has_any (protocols)), TargetInfoProtocol, 'NotApplicable')
| summarize StartTime = min(TimeGenerated), EndTime = max(TimeGenerated), AccountsUsedCount =
dcount(TargetAccount), AccountsUsed = make_set(TargetAccount), TargetMachineCount = dcount(TargetInfoMachine),
TargetMachines = make_set(TargetInfoMachine), TargetProtocols = dcount(TargetInfoProtocol), Protocols =
make_set(TargetInfoProtocol), Processes = make_set(Process) by Computer, SubjectAccount, UncommonProtocol
| where TargetMachineCount > 1 or UncommonProtocol != 'NotApplicable'
| extend ProtocolCount = array_length(Protocols)
| extend ProtocolScore = case(
Protocols has 'rpcss' and Protocols has 'host' and Protocols has 'cifs', 10, //observed in Solorigate and depending on which
are used together the higher the score
Protocols has 'rpcss' and Protocols has 'host', 5,
Protocols has 'rpcss' and Protocols has 'cifs', 5,
Protocols has 'host' and Protocols has 'cifs', 5,
Protocols has 'ldap' or Protocols has 'rpcss' or Protocols has 'host' or Protocols has 'cifs', 1, //ldap is more commonly seen in
general, this was also seen with Solorigate but not usually to the same machines as the others above
UncommonProtocol != 'NotApplicable', 3,
0 //other protocols may be of interest, but in relation to observations for enumeration/execution in Solorigate they receive 0
)
| extend Score = ProtocolScore + ProtocolCount + AccountsUsedCount
| where Score >= 9 or (UncommonProtocol != 'NotApplicable' and Score >= 4) // Score must be 9 or better as this will
include 5 points for atleast 2 of the interesting protocols + the count of protocols (min 2) + the number of accounts used for
execution (min 2) = min of 9 OR score must be 4 or greater for an uncommon protocol
| extend TimePeriod = EndTime - StartTime //This identifies the time between start and finish for the use of the explicit
```

```
credentials, shorter time period may indicate scripted executions
| project-away UncommonProtocol
| extend timestamp = StartTime, AccountCustomEntity = SubjectAccount, HostCustomEntity = Computer
| order by Score desc\n\n
```

Data Access

\n

Accessing confidential data is one of the primary motives of this attack. Data access for the attacker here relied on leveraging minted SAML tokens to access user files/email stored in the cloud via compromised AppIds. One way to detect this is when a user or application signs in using [Azure Active Directory PowerShell to access non-Active Directory resources](#).

\n\n

Microsoft Graph is one way that the attacker may be seen accessing resources and can help find what the attacker may have accessed using the Service principal Azure Active Directory sign-in logs. If you have data in your Log analytics you could easily plot a chart to see what anomalous activity is happening in your environment that is leveraging the graph.

\n

Updated 12/17/2020

\n

Note that this data type in Azure Sentinel below is only available when additional Diagnostic Logging is enabled on the workspace. Please see the instructions in the expandable section below.

\nThe AADServicePrincipalSignInLogs datatype will not be available in Azure Sentinel unless it is configured under Diagnostic Settings. Please see screenshots below the query.

```
AADServicePrincipalSignInLogs
| where TimeGenerated > ago(90d)
| where ResourceDisplayName == \"Microsoft Graph\"
| where ServicePrincipalId == \"524c43c4-c484-4f7a-bd44-89d4a0d8aeab\"
| summarize count() by bin(TimeGenerated, 1h)
| render timechart
```

To enable Service Principal Signin Logging, do the following:

\n\n\n\n\n\n\n

Updated 12/21/2020

\n

Additionally, below is a sample query that brings out some of the logons to Azure AD where multi factor authentication was satisfied by token based logons versus MFA via phone auth or the like. It is possible this could produce many results, so additional tuning is suggested for your environment.

```
\nSignInLogs
| where TimeGenerated > ago(30d)
| where ResultType == 0
| extend additionalDetails = tostring(Status.additionalDetails)
| summarize make_set(additionalDetails), min(TimeGenerated), max(TimeGenerated) by IPAddress, UserPrincipalName
| where array_length(set_additionalDetails) == 2
| where (set_additionalDetails[1] == \"MFA requirement satisfied by claim in the token\" and set_additionalDetails[0] == \"MFA requirement satisfied by claim provided by external provider\") or (set_additionalDetails[0] == \"MFA requirement satisfied by claim in the token\" and set_additionalDetails[1] == \"MFA requirement satisfied by claim provided by external provider\")
// project IPAddress, UserPrincipalName, min_TimeGenerated, max_TimeGenerated\n
```

UPDATED 12/17/2020

\n

This attack also used Virtual Private Servers (VPS) hosts to access victim networks and can be used in conjunction with the query above. Both MSTIC and FireEye have reported attacker logon events coming from network ranges associated with VPS providers. In order to highlight these logons, MSTIC has created a new hunting query - [Signins From VPS Providers](#) - that looks for successful signins from network ranges associated with VPS providers. This is joined with the above query,

the new query looks for IPs that only display sign-ins based on tokens and not other MFA options, although this could be removed if wanted. The list of VPS ranges in the query is not comprehensive and there is significant potential for false positives so results should be investigated before responding, however it can provide very effective signal. Combining the query below with data that list VPS server ranges will make this a high-confidence hunting query.

\n\n

In relation to the VPS servers section above, the previously mentioned workbook has a section that shows successful user signins from VPS (Virtual Private Server) providers where only tokens were used to authenticate. This uses the new KQL operator [jpy4_lookup](#) to evaluate if a login came from a known VPS provider network range. This operator can alternatively be used to look for all logons not coming from known ranges should your environment have a common logon source.

\n\n

Data Exfiltration

\n

Updated 12/20/2020

\n

Email data has been observed as a target for the Solorigate attackers, one way to monitor for potential suspicious access is to look for anomalous MailItemsAccessed volumes. MSTIC has created a specific hunting query to identify [Anomalous User Accessing Other Users Mailbox](#) which can help to identify malicious activity related to this attack. Additionally, MSTIC previously created a more generic detection - [Exchange workflow MailItemsAccessed operation anomaly](#) - which looks for time series based anomalies in MailItemsAccessed events in the OfficeActivity log.

\nAnomalous access to other user's mailboxes

```
let timeframe = 14d;
let user_threshold = 1;
let folder_threshold = 5;
OfficeActivity
| where TimeGenerated > ago(timeframe)
| where Operation =~ \"MailItemsAccessed\"
| where ResultStatus =~ \"Succeeded\"
| mv-expand parse_json(Folders)
| extend folders = tostring(Folders.Path)
| where tolower(MailboxOwnerUPN) != tolower(UserId)
| extend ClientIP = iif(Client_IPAddress startswith \"[\", extract(\"\\\\\\\\\\\\\\\\[{}\\\\\\\\\\\\\\\\]*\"), 1, Client_IPAddress), Client_IPAddress)
| summarize make_set(folders), make_set(ClientInfoString), make_set(ClientIP), make_set(MailboxGuid),
make_set(MailboxOwnerUPN) by UserId
| extend folder_count = array_length(set_folders)
| extend user_count = array_length(set_MailboxGuid)
| where user_count > user_threshold or folder_count > folder_threshold
| sort by user_count desc
| project-reorder UserId, user_count, folder_count, set_MailboxOwnerUPN, set_ClientIP, set_ClientInfoString, set_folder
```

Exchange workflow MailItemsAccessed operation anomaly

```
let starttime = 14d;
let endtime = 1d;
let timeframe = 1h;
let scorethreshold = 1.5;
let percentthreshold = 50;
// Preparing the time series data aggregated hourly count of MailItemsAccessd Operation in the form of multi-value array to
use with time series anomaly function.
let TimeSeriesData =
OfficeActivity
| where TimeGenerated between (startofday(ago(starttime))).startofday(ago(endtime)))
| where OfficeWorkload =~ \"Exchange\" and Operation =~ \"MailItemsAccessed\" and ResultStatus =~ \"Succeeded\"
| project TimeGenerated, Operation, MailboxOwnerUPN
| make-series Total=count() on TimeGenerated from startofday(ago(starttime)) to startofday(ago(endtime)) step timeframe;
let TimeSeriesAlerts = TimeSeriesData
| extend (anomalies, score, baseline) = series_decompose_anomalies(Total, scorethreshold, -1, 'linefit')
| mv-expand Total to typeof(double), TimeGenerated to typeof(datetime), anomalies to typeof(double), score to
typeof(double), baseline to typeof(long)
| where anomalies > 0
```

```
| project TimeGenerated, Total, baseline, anomalies, score;
// Joining the flagged outlier from the previous step with the original dataset to present contextual information
// during the anomalyhour to analysts to conduct investigation or informed decisions.
TimeSeriesAlerts | where TimeGenerated > ago(2d)
// Join against base logs since specified timeframe to retrieve records associated with the hour of anomaly
| join (
OfficeActivity
| where TimeGenerated > ago(2d)
| where OfficeWorkload=~ "Exchange" and Operation =~ "MailItemsAccessed" and ResultStatus =~ "Succeeded"
) on TimeGenerated
```

\n

Updated 12/19/2020

\n

Targeting of email data has also been observed by other industry members including [Volexity](#) who reported attackers using PowerShell commands to export on premise Exchange mailboxes and then hosting those files on OWA servers in order to exfiltrate them.

\n

MSTIC has created detections to identify this activity at both the [OWA server](#) and [attacking host level](#) through IIS logs, and PowerShell command line logging.

\n\n

OWA exfiltration:

```
\nlet excludeIps = dynamic(["127.0.0.1", "::1"]);
let scriptingExt = dynamic([".aspx", ".ashx", ".asp"]);
W3CIISLog
| where csUriStem contains "/owa/"
//The actor pulls a file back but won't send it any URI params
| where isempty(csUriQuery)
| extend file_ext = tostring(split(csUriStem, "\\")[1])
//Giving your file a known scripting extension will throw an error
//rather than just serving the file as it will try to interpret the script
| where file_ext !in~ (scriptingExt)
//The actor was seen using image files, but we go wider in case they change this behaviour
//| where file_ext in~ (".jpg", ".jpeg", ".png", ".bmp")
| extend file_name = tostring(split(csUriStem, "\\")[1])
| where file_name != ""
| where cIP !in~ (excludeIps)
| project file_ext, csUriStem, file_name, Computer, cIP, sIP, TenantId, TimeGenerated
| summarize dcount(cIP), AccessingIPs=make_set(cIP), AccessTimes=make_set(TimeGenerated), Access=count() by
TenantId, file_name, Computer, csUriStem
//Collection of the exfiltration will occur only once, lets check for 2 accesses in case they mess up
//Tailor this for hunting
| where Access <= 2 and dcount_cIP == 1\n
```

Host creating then removing mailbox export requests using PowerShell cmdlets:

\n\n

```
// Adjust the timeframe to change the window events need to occur within to alert
```

\n

```
let timeframe = 1h;
```

\n

```
SecurityEvent
```

\n

```
| where Process in ("powershell.exe", "cmd.exe")
```

```
\n
| where CommandLine contains 'New-MailboxExportRequest'
\n
| summarize by Computer, timekey = bin(TimeGenerated, timeframe), CommandLine, SubjectUserName
\n
| join kind=inner (SecurityEvent
\n
| where Process in ('powershell.exe','cmd.exe'))
\n
| where CommandLine contains 'Remove-MailboxExportRequest'
\n
| summarize by Computer, timekey = bin(TimeGenerated, timeframe), CommandLine, SubjectUserName) on Computer,
timekey, SubjectUserName
\n
| extend commands = pack_array(CommandLine1, CommandLine)
\n
| summarize by timekey, Computer, tostring(commands), SubjectUserName
\n
| project-reorder timekey, Computer, SubjectUserName, ['commands']
\n
| extend HostCustomEntity = Computer, AccountCustomEntity = SubjectUserName
\n\n
```

Updated 12/28/2020

\n

Email Delegation and later delegate access is another tactic that has been observed to gain access to user's mailboxes. We have a previously created a method to discover [Non-owner mailbox login activity](#) that can be applied here to help identify when delegates are inappropriately access email.

```
\n\nlet timeframe = 1d;
OfficeActivity
| where TimeGenerated >= ago(timeframe)
| where Operation == \"MailboxLogin\" and Logon_Type != \"Owner\"
| summarize count(), min(TimeGenerated), max(TimeGenerated) by Operation, OrganizationName, UserType, UserId,
MailboxOwnerUPN, Logon_Type
| extend timestamp = min_TimeGenerated, AccountCustomEntity = UserId\n\n
```

Domain Hunting

\n

Updated 12/17/2020

\n

Domain specific

\n

MSTIC has collated network based IoCs from [MSTIC](#), [FireEye](#) and [Volatility](#) to create a network based IoC detection - [Solorigate Network Beacon](#) - that leverage multiple network focused data sources within Azure Sentinel.

```

\inlet domains =
dynamic(["incomupdate.com","zupertech.com","databasegalore.com","panhardware.com","avsvmcloud.com","digitalcollege.org","freescanonli
let timeframe = 6h;
(union isfuzzy=true
(CommonSecurityLog
| where TimeGenerated >= ago(timeframe)
| parse Message with * (' DNSName ')*
| where DNSName in~ (domains) or DestinationHostName has_any (domains) or RequestURL has_any(domains)
| extend AccountCustomEntity = SourceUserID, HostCustomEntity = DeviceName, IPCustomEntity = SourceIP
),
(DnsEvents
| where TimeGenerated >= ago(timeframe)
| extend DNSName = Name
| where isnotempty(DNSName)
| where DNSName in~ (domains)
| extend IPCustomEntity = ClientIP
),
(VMConnection
| where TimeGenerated >= ago(timeframe)
| parse RemoteDnsCanonicalNames with * '[' DNSName ']' *
| where isnotempty(DNSName)
| where DNSName in~ (domains)
| extend IPCustomEntity = RemoteIp
),
(DeviceNetworkEvents
| where TimeGenerated >= ago(timeframe)
| where isnotempty(RemoteUrl)
| where RemoteUrl has_any (domains)
| extend DNSName = RemoteUrl
| extend IPCustomEntity = RemoteIP
| extend HostCustomEntity = DeviceName
)
)n

```

Domain DGA

\n

The avsvmcloud[.]com has been observed by several organizations as making DGA like subdomain queries as part of C2 activities. MSTIC have generated a hunting query - [Solorigate DNS Pattern](#) - to look for similar patterns of activity from other domains that might help identify other potential C2 sources.

```

\inlet cloudApiTerms = dynamic(["api", "east", "west"]);
DnsEvents
| where IPAddresses != "" and IPAddresses != "127.0.0.1"
| where Name endswith ".com" or Name endswith ".org" or Name endswith ".net"
| extend domain_split = split(Name, ".")
| where tostring(domain_split[-5]) != "" and tostring(domain_split[-6]) == ""
| extend sub_domain = tostring(domain_split[0])
| where sub_domain !contains "-"
| extend sub_directories = strcat(domain_split[-3], ".", domain_split[-4])
| where sub_directories has_any(cloudApiTerms)
//Based on sample communications the subdomain is always between 20 and 30 bytes
| where strlen(domain_split) < 32 or strlen(domain_split) > 20
| extend domain = strcat(tostring(domain_split[-2]), ".", tostring(domain_split[-1]))
| extend subdomain_no = countof(sub_domain, @"(\d)", "regex")
| extend subdomain_ch = countof(sub_domain, @"([a-z])", "regex")
| where subdomain_no > 1
| extend percentage_numerical = toreal(subdomain_no) / toreal(strlen(sub_domain)) * 100
| where percentage_numerical < 50 and percentage_numerical > 5
| summarize count(), make_set(Name), FirstSeen=min(TimeGenerated), LastSeen=max(TimeGenerated) by Name
| order by count_ asc\n

```

Encoded Domain

\n

In addition we have another query - [Solorigate Encoded Domain in URL](#)- that takes the encoding pattern the DGA uses, encodes the domains seen in signin logs and then looks for those patterns in DNS logs. This can help identify other C2 domains using the same encoding scheme.

```
\nlet dictionary =
dynamic([\r\n,q,\"3\",g,\"s\",a,\"l\",t\",6\",u\",1\",i\",y\",f\",z\",o\",p\",5\",7\",2\",d\",4\",9\",b\",n\",x\",8\",c]
let regex_bad_domains = SigninLogs
//Collect domains from tenant from signin logs
| where TimeGenerated > ago(1d)
| extend domain = tostring(split(UserPrincipalName, \"@\", 1)[0])
| where domain != \"\"
| summarize by domain
| extend split_domain = split(domain, \".\")
//This cuts back on domains such as na.contoso.com by electing not to match on the \"na\" portion
| extend target_string = iff(strlen(split_domain[0]) <= 2, split_domain[1], split_domain[0])
| extend target_string = split(target_string, \"-\")
| mv-expand target_string
//Rip all of the alphanumeric out of the domain name
| extend string_chars = extract_all(@(\"([a-z0-9])\"), tostring(target_string))
//Guid for tracking our data
| extend guid = new_guid()
//Expand to get all of the individual chars from the domain
| mv-expand string_chars
| extend chars = tostring(string_chars)
//Conduct computation to encode the domain as per actor spec
| extend computed_char = array_index_of(dictionary, chars)
| extend computed_char = dictionary[(computed_char + 4) % array_length(dictionary)]
| summarize make_list(computed_char) by guid, domain
| extend target_encoded = tostring(strcat_array(list_computed_char, \"\"))
//These are probably too small, but can be edited (expect FP's when going too small)
| where strlen(target_encoded) > 5
| distinct target_encoded
| summarize make_set(target_encoded)
//Key to join to DNS
| extend key = 1;
DnsEvents
| where TimeGenerated > ago(1d)
| summarize by Name
| extend key = 1
//For each DNS query join the malicious domain list
| join kind=inner (
regex_bad_domains
) on key
| project-away key
//Expand each malicious key for each DNS query observed
| mv-expand set_target_encoded
//IndexOf allows us to fuzzy match on the substring
| extend match = indexof(Name, set_target_encoded)
| where match > -1\n
```

Security Service Tampering

\n

Updated 01/19/2021

\n

There has been additional indication that security services are being tampered with to hinder detection and investigation. While this is a common tactic, we felt that we should include this reference. The query is currently written specifically for [Potential Microsoft security services tampering](#), but can easily be adapted to identify other security services.

```
\nlet includeProc = dynamic([\sc.exe\",net1.exe\",net.exe\",taskkill.exe\",cmd.exe\",powershell.exe\"]);
let action = dynamic([\stop\",disable\",delete\"]);
```

```

let service1 = dynamic(['sense', 'windefend', 'mssecflt']);
let service2 = dynamic(['sense', 'windefend', 'mssecflt', 'healthservice']);
let params1 = dynamic(['-DisableRealtimeMonitoring', '-DisableBehaviorMonitoring', '-DisableIOAVProtection']);
let params2 = dynamic(['sgrmbroker.exe', 'mssense.exe']);
let regparams1 = dynamic(['reg add "HKLM\SOFTWARE\Policies\Microsoft\Windows Defender", reg add
"HKLM\SOFTWARE\Policies\Microsoft\Windows Advanced Threat Protection"]);
let regparams2 = dynamic(['ForceDefenderPassiveMode', 'DisableAntiSpyware']);
let regparams3 = dynamic(['sense', 'windefend']);
let regparams4 = dynamic(['demand', 'disabled']);
let regparams5 = dynamic(['reg add "HKLM\SYSTEM\CurrentControlSet\services\HealthService", reg add
"HKLM\SYSTEM\CurrentControlSet\Services\Sense", reg add
"HKLM\SYSTEM\CurrentControlSet\Services\WinDefend", reg add
"HKLM\SYSTEM\CurrentControlSet\Services\MsSecFlt", reg add
"HKLM\SYSTEM\CurrentControlSet\Services\DiagTrack", reg add
"HKLM\SYSTEM\CurrentControlSet\Services\SgrmBroker", reg add
"HKLM\SYSTEM\CurrentControlSet\Services\SgrmAgent", reg add
"HKLM\SYSTEM\CurrentControlSet\Services\AATPSensorUpdater", reg add
"HKLM\SYSTEM\CurrentControlSet\Services\AATPSensor", reg add
"HKLM\SYSTEM\CurrentControlSet\Services\mpssvc"]);
let regparams6 = dynamic(['/d 4;/d "4"/d 0x00000004']);
let regparams7 = dynamic(['/d 1;/d "1"/d 0x00000001']);
let timeframe = 1d;
(union isfuzzy=true
(
SecurityEvent
| where TimeGenerated >= ago(timeframe)
| where EventID == 4688
| extend ProcessName = tostring(split(NewProcessName, '\\')[1])
| where ProcessName in~ (includeProc)
| where (CommandLine has_any (action) and CommandLine has_any (service1))
or (CommandLine has_any (params1) and CommandLine has 'Set-MpPreference' and CommandLine has 'true')
or (CommandLine has_any (params2) and CommandLine has '/IM')
or (CommandLine has_any (regparams5) and CommandLine has 'Start' and CommandLine has_any (regparams6))
or (CommandLine has_any (regparams1) and CommandLine has_any (regparams2) and CommandLine has_any
(regparams7))
or (CommandLine has 'start' and CommandLine has 'config' and CommandLine has_any (regparams3) and
CommandLine has_any (regparams4))
| project TimeGenerated, Computer, Account, AccountDomain, ProcessName, ProcessNameFullPath = NewProcessName,
EventID, Activity, CommandLine, EventSourceName, Type
),
(
Event
| where TimeGenerated >= ago(timeframe)
| where Source =~ "Microsoft-Windows-SENSE"
| where EventID == 87 and ParameterXml in ("<Param>sgrmbroker</Param>", "<Param>WinDefend</Param>")
| project TimeGenerated, Computer, Account = UserName, EventID, Activity = RenderedDescription, EventSourceName =
Source, Type
),
(
DeviceProcessEvents
| where TimeGenerated >= ago(timeframe)
| where InitiatingProcessFileName in~ (includeProc)
| where (InitiatingProcessCommandLine has_any(action) and InitiatingProcessCommandLine has_any (service2) and
InitiatingProcessParentFileName != 'cscript.exe')
or (InitiatingProcessCommandLine has_any (params1) and InitiatingProcessCommandLine has 'Set-MpPreference' and
InitiatingProcessCommandLine has 'true')
or (InitiatingProcessCommandLine has_any (params2) and InitiatingProcessCommandLine has '/IM')
or (InitiatingProcessCommandLine has_any (regparams5) and InitiatingProcessCommandLine has 'Start' and
InitiatingProcessCommandLine has_any (regparams6))
or (InitiatingProcessCommandLine has_any (regparams1) and InitiatingProcessCommandLine has_any (regparams2) and
InitiatingProcessCommandLine has_any (regparams7))
or (InitiatingProcessCommandLine has_any("start") and InitiatingProcessCommandLine has 'config' and
InitiatingProcessCommandLine has_any (regparams3) and InitiatingProcessCommandLine has_any (regparams4))
| extend Account = iff(isnotempty(InitiatingProcessAccountUpn), InitiatingProcessAccountUpn,

```


New user created and added to the built-in administrators group	https://github.com/Azure/Azure-Sentinel/blob/master/Detections/SecurityEvent/UserCreatedAddedToBuiltinAdmins_1d.yaml
User account added to built in domain local or global group	https://github.com/Azure/Azure-Sentinel/blob/master/Detections/SecurityEvent/UserAccountAddedToPrivilegeGroup_1h.yaml
Tracking Privileged Account Rare Activity	https://github.com/Azure/Azure-Sentinel/blob/master/Hunting%20Queries/MultipleDataSources/TrackingPrivAccounts.yaml
ADFS Key Extraction	
ADFS DKM Master Key Export	https://github.com/Azure/Azure-Sentinel/blob/master/Detections/MultipleDataSources/ADFS-DKM-MasterKey-Export.yaml
ADFS Key Export (Sysmon)	https://github.com/Azure/Azure-Sentinel/blob/master/Detections/SecurityEvent/ADFSKeyExportSysmon.yaml
Entropy for Processes for a given Host	https://github.com/Azure/Azure-Sentinel/blob/master/Hunting%20Queries/SecurityEvent/ProcessEntropy.yaml
Rare processes run by Service accounts	https://github.com/Azure/Azure-Sentinel/blob/master/Hunting%20Queries/SecurityEvent/RareProcbByServiceAccount.yaml
Uncommon processes - bottom 5%	https://github.com/Azure/Azure-Sentinel/blob/master/Hunting%20Queries/SecurityEvent/uncommon_processes.yaml
Azure Active Directory	
Modified domain federation trust settings	https://github.com/Azure/Azure-Sentinel/blob/master/Detections/AuditLogs/ADFSDomainTrustMods.yaml
New access credential added to Application or Service Principal	https://github.com/Azure/Azure-Sentinel/blob/master/Detections/AuditLogs/NewAppOrServicePrincipalCredential.yaml
First access credential added to Application or Service Principal where no credential was present	https://github.com/Azure/Azure-Sentinel/blob/master/Detections/AuditLogs/FirstAppOrServicePrincipalCredential.yaml
Mail.Read Permissions Granted to Application	https://github.com/Azure/Azure-Sentinel/blob/master/Detections/AuditLogs/MailPermissionsAddedToApplication.yaml
Suspicious application consent similar to O365 Attack Toolkit	https://github.com/Azure/Azure-Sentinel/blob/master/Detections/AuditLogs/MaliciousOAuthApp_O365AttackToolkit.yaml
Suspicious application consent for offline access	https://github.com/Azure/Azure-Sentinel/blob/master/Detections/AuditLogs/SuspiciousOAuthApp_OfflineAccess.yaml

Recon and Remote Execution	
Suspicious enumeration using Adfind tool	https://github.com/Azure/Azure-Sentinel/blob/master/Hunting%20Queries/SecurityEvent/Suspicious_enumeration_using_adfind.yaml
Multiple explicit credential usage - 4648 events	https://github.com/Azure/Azure-Sentinel/blob/master/Hunting%20Queries/SecurityEvent/MultipleExplicitCredentialUsage4648Events.yaml
Data Access	
Azure Active Directory PowerShell accessing non-AAD resources	https://github.com/Azure/Azure-Sentinel/blob/master/Detections/SigninLogs/AzureAADPowerShellAnomaly.yaml
Signins From VPS Providers	https://github.com/Azure/Azure-Sentinel/blob/master/Hunting%20Queries/SigninLogs/Signins-From-VPS-Providers.yaml
Data Exfiltration	
Anomalous access to other user's mailboxes	https://github.com/Azure/Azure-Sentinel/blob/master/Hunting%20Queries/OfficeActivity/AnomalousUserAccessingOtherUsersMailbox.yaml
Exchange workflow MailItemsAccessed operation anomaly	https://github.com/Azure/Azure-Sentinel/blob/master/Detections/OfficeActivity/MailItemsAccessedTimeSeries.yaml
Suspect Mailbox Export on IIS/OWA	https://github.com/Azure/Azure-Sentinel/blob/master/Hunting%20Queries/W3CIISLog/SuspectedMailBoxExportHostonOWA.yaml
Host Exporting Mailbox and Removing Export	https://github.com/Azure/Azure-Sentinel/blob/master/Hunting%20Queries/SecurityEvent/HostExportingMailboxAndRemovingExport.yaml
Non-owner mailbox login activity	https://github.com/Azure/Azure-Sentinel/blob/master/Hunting%20Queries/OfficeActivity/nonowner_MailboxLogin.yaml
Domain Hunting	
Solorigate Network Beacon	https://github.com/Azure/Azure-Sentinel/blob/master/Detections/MultipleDataSources/Solorigate-Network-Beacon.yaml
Solorigate DNS Pattern	https://github.com/Azure/Azure-Sentinel/blob/master/Hunting%20Queries/DnsEvents/Solorigate-DNS-Pattern.yaml
Solorigate Encoded Domain in URL	https://github.com/Azure/Azure-Sentinel/blob/master/Hunting%20Queries/DnsEvents/Solorigate-Encoded-Domain-URL.yaml
Security Service Tampering	
Potential Microsoft security services tampering	https://github.com/Azure/Azure-Sentinel/blob/master/Hunting%20Queries/MultipleDataSources/PotentialMicrosoftSecurityServicesTampering.yaml
M365+Sentinel	
Solorigate Defender Detections	https://github.com/Azure/Azure-Sentinel/blob/master/Detections/SecurityAlert/Solorigate-Defender-Detections.yaml

\n\n

References


```
{\"url\": \"https://techcommunity.microsoft.com/t5/s/gxcuf89792/images/bS0xOTk1MDk1LTI0MTUyNm00QUEzQjFDRjM0MkI0OTU4?revision=57\"}}, {\"__typename\": \"AssociatedImageEdge\", \"cursor\": \"MjYuMXwyLjF8b3wyNXxfTIZffDQ\", \"node\": {\"__ref\": \"AssociatedImage\": {\"url\": \"https://techcommunity.microsoft.com/t5/s/gxcuf89792/images/bS0xOTk1MDk1LTI0MTUyN2IBRTZGNDQ4NjhDMUFCNDM5?revision=57\"}}, {\"__typename\": \"AssociatedImageEdge\", \"cursor\": \"MjYuMXwyLjF8b3wyNXxfTIZffDU\", \"node\": {\"__ref\": \"AssociatedImage\": {\"url\": \"https://techcommunity.microsoft.com/t5/s/gxcuf89792/images/bS0xOTk1MDk1LTI0MTUyOGIDQjYxQ0JGRVCVCNjE3MUQ3?revision=57\"}}}}, {\"totalCount\": 5, \"pageInfo\": {\"__typename\": \"PageInfo\", \"hasNextPage\": false, \"endCursor\": null, \"hasPreviousPage\": false, \"startCursor\": null}}, \"attachments\": {\"__typename\": \"AttachmentConnection\", \"pageInfo\": {\"__typename\": \"PageInfo\", \"hasNextPage\": false, \"endCursor\": null, \"hasPreviousPage\": false, \"startCursor\": null}}, \"edges\": [], \"tags\": {\"__typename\": \"TagConnection\", \"pageInfo\": {\"__typename\": \"PageInfo\", \"hasNextPage\": false, \"endCursor\": null, \"hasPreviousPage\": false, \"startCursor\": null}}, \"edges\": [{\"__typename\": \"TagEdge\", \"cursor\": \"MjYuMXwyLjF8b3wxMHxfTIZffDE\", \"node\": {\"__typename\": \"Tag\", \"id\": \"tag:hunting\", \"text\": \"hunting\", \"time\": \"2019-04-11T09:00:00.012-07:00\", \"lastActivityTime\": null, \"messagesCount\": null, \"followersCount\": null}}, {\"__typename\": \"TagEdge\", \"cursor\": \"MjYuMXwyLjF8b3wxMHxfTIZffDI\", \"node\": {\"__typename\": \"Tag\", \"id\": \"tag:microsoft sentinel\", \"text\": \"microsoft sentinel\", \"time\": \"2021-11-02T10:33:48.383-07:00\", \"lastActivityTime\": null, \"messagesCount\": null, \"followersCount\": null}}}], \"timeToRead\": 33, \"rawTeaser\": \"\n
```

Microsoft recently blogged about the [Recent Nation-State Cyber Attacks](#) that has impacted high value targets both across the government and private sector. This attack is also known as [Solorigate](#) or [Sunburst](#).

```
\"introduction\": \"\", \"coverImage\": null, \"coverImageProperties\": {\"__typename\": \"CoverImageProperties\", \"style\": \"STANDARD\", \"titlePosition\": \"BOTTOM\", \"altText\": \"\", \"currentRevision\": {\"__ref\": \"Revision:revision:1995095_57\"}, \"latestVersion\": {\"__typename\": \"FriendlyVersion\", \"major\": \"55\", \"minor\": \"0\"}, \"metrics\": {\"__typename\": \"MessageMetrics\", \"views\": 103654, \"read\": false, \"visibilityScope\": \"PUBLIC\", \"canonicalUrl\": null, \"seoTitle\": null, \"seoDescription\": null, \"edges\": {\"__typename\": \"UserConnection\", \"edges\": [], \"nonCoAuthorContributors\": {\"__typename\": \"UserConnection\", \"edges\": [], \"coAuthors\": {\"__typename\": \"UserConnection\", \"edges\": []}, \"blogMessagePolicies\": {\"__typename\": \"BlogMessagePolicies\", \"canDoAuthoringActionsOnBlog\": {\"__typename\": \"PolicyResult\", \"failureReason\": {\"__typename\": \"FailureReason\", \"message\": \"error.lithium.policies.blog.action_can_do_authoring_action.accessDenied\", \"key\": \"error.lithium.policies.blog\"}}, \"archivalData\": null, \"customFields\": [], \"revisions\": {\"constraints\": {\"isPublished\": {\"eq\": true}}}}}, {\"__typename\": \"RevisionConnection\", \"totalCount\": 57}}, \"Conversation:conversation:1995095\": {\"__typename\": \"Conversation\", \"id\": \"conversation:1995095\", \"solved\": false, \"topic\": {\"__ref\": \"BlogTopicMessage:message:1995095\"}, \"lastPostingActivityTime\": \"2021-11-02T18:30:13.281-07:00\", \"lastPostTime\": \"2020-12-27T00:51:08.223-08:00\", \"unreadReplyCount\": 5, \"isSubscribed\": false}, \"ModerationData:moderation_data:1995095\": {\"__typename\": \"ModerationData\", \"id\": \"moderation_data:1995095\", \"status\": \"APPROVED\", \"rejectReason\": null, \"isReportedAbuse\": false, \"rejectUser\": null}, {\"url\": \"https://techcommunity.microsoft.com/t5/s/gxcuf89792/images/bS0xOTk1MDk1LTI0MTE2OGIFQjcxOEM5Njc1QzQ5Q0Q3?revision=57\"}: {\"__typename\": \"AssociatedImage\", \"url\": \"https://techcommunity.microsoft.com/t5/s/gxcuf89792/images/bS0xOTk1MDk1LTI0MTE2OGIFQjcxOEM5Njc1QzQ5Q0Q3?revision=57\", \"title\": \"Image1.png\", \"associationType\": \"TEASER\", \"width\": 799, \"height\": 527, \"altText\": null}, \"AssociatedImage\": {\"url\": \"https://techcommunity.microsoft.com/t5/s/gxcuf89792/images/bS0xOTk1MDk1LTI0MTE1MWkzRkU2NThDUY5QzZmZmNtG1?revision=57\"}: {\"__typename\": \"AssociatedImage\", \"url\": \"https://techcommunity.microsoft.com/t5/s/gxcuf89792/images/bS0xOTk1MDk1LTI0MTE1MWkzRkU2NThDUY5QzZmZmNtG1?revision=57\", \"title\": \"Image1.png\", \"associationType\": \"BODY\", \"width\": 799, \"height\": 527, \"altText\": null}, \"AssociatedImage\": {\"url\": \"https://techcommunity.microsoft.com/t5/s/gxcuf89792/images/bS0xOTk1MDk1LTI0MTUyNm00QUEzQjFDRjM0MkI0OTU4?revision=57\"}: {\"__typename\": \"AssociatedImage\", \"url\": \"https://techcommunity.microsoft.com/t5/s/gxcuf89792/images/bS0xOTk1MDk1LTI0MTUyNm00QUEzQjFDRjM0MkI0OTU4?revision=57\", \"title\": \"DiagnosticSettings1.png\", \"associationType\": \"BODY\", \"width\": 3454, \"height\": 1879, \"altText\": null}, \"AssociatedImage\": {\"url\": \"https://techcommunity.microsoft.com/t5/s/gxcuf89792/images/bS0xOTk1MDk1LTI0MTUyN2IBRTZGNDQ4NjhDMUFCNDM5?revision=57\"}: {\"__typename\": \"AssociatedImage\", \"url\": \"https://techcommunity.microsoft.com/t5/s/gxcuf89792/images/bS0xOTk1MDk1LTI0MTUyN2IBRTZGNDQ4NjhDMUFCNDM5?revision=57\", \"title\": \"DiagnosticSettings2.png\", \"associationType\": \"BODY\", \"width\": 3679, \"height\": 1006, \"altText\": null}, \"AssociatedImage\": {\"url\": \"https://techcommunity.microsoft.com/t5/s/gxcuf89792/images/bS0xOTk1MDk1LTI0MTUyOGIDQjYxQ0JGRVCVCNjE3MUQ3?revision=57\"}: {\"__typename\": \"AssociatedImage\", \"url\": \"https://techcommunity.microsoft.com/t5/s/gxcuf89792/images/bS0xOTk1MDk1LTI0MTUyOGIDQjYxQ0JGRVCVCNjE3MUQ3?revision=57\", \"title\": \"DiagnosticSettings3.png\", \"associationType\": \"BODY\", \"width\": 2040, \"height\": 1318, \"altText\": null}, \"Revision:revision:1995095_57\": {\"__typename\": \"Revision\", \"id\": \"revision:1995095_57\", \"lastEditTime\": \"2021-11-02T18:30:13.281-07:00\", \"CachedAsset:theme:customTheme1-1775107807370\": {\"__typename\": \"CachedAsset\", \"id\": \"theme:customTheme1-1775107807370\", \"value\": {\"id\": \"customTheme1\", \"animation\": {\"fast\": \"150ms\", \"normal\": \"250ms\", \"slow\": \"500ms\", \"slowest\": \"750ms\", \"function\": \"cubic-bezier(0.07, 0.91, 0.51, 1)\", \"__typename\": \"AnimationThemeSettings\"}, \"avatar\": {\"borderRadius\": \"50%\", \"collections\":
```

```
[ "default" ], "__typename": "AvatarThemeSettings" }, "basics": { "browserIcon": { "imageAssetName": "favicon-1730836283320.png", "imageLastModified": "1730836286415", "__typename": "ThemeAsset" }, "customerLogo": { "imageAssetName": "favicon-1730836271365.png", "imageLastModified": "1730836274203", "__typename": "ThemeAsset" }, "maximumWidthOfPageContent": "1300px", "oneColumn": { "borderRadiusSm": "3px", "borderRadius": "3px", "borderRadiusLg": "5px", "paddingY": "5px", "paddingYLg": "7px", "paddingYHero": "var(--lia-bs-btn-padding-y-lg)", "paddingX": "12px", "paddingXLg": "16px", "paddingXHero": "60px", "fontStyle": "NORMAL", "fontWeight": "700", "textTransform": "NONE", "disabled": "var(--lia-bs-white)", "primaryTextHoverColor": "var(--lia-bs-white)", "primaryTextActiveColor": "var(--lia-bs-white)", "primaryBgColor": "var(--lia-bs-primary)", "primaryBgHoverColor": "hsl(var(--lia-bs-primary-h), var(--lia-bs-primary-s), calc(var(--lia-bs-primary-l) * 0.85))", "primaryBgActiveColor": "hsl(var(--lia-bs-primary-h), var(--lia-bs-primary-s), calc(var(--lia-bs-primary-l) * 0.7))", "primaryBorder": "1px solid transparent", "primaryBorderHover": "1px solid transparent", "primaryBorderActive": "1px solid transparent", "primaryBorderFocus": "1px solid var(--lia-bs-white)", "primaryBoxShadowFocus": "0 0 1px var(--lia-bs-primary), 0 0 4px hsla(var(--lia-bs-primary-h), var(--lia-bs-primary-s), var(--lia-bs-primary-l), 0.2)", "secondaryTextColor": "var(--lia-bs-gray-900)", "secondaryTextHoverColor": "hsl(var(--lia-bs-gray-900-h), var(--lia-bs-gray-900-s), calc(var(--lia-bs-gray-900-l) * 0.95))", "secondaryTextActiveColor": "hsl(var(--lia-bs-gray-900-h), var(--lia-bs-gray-900-s), calc(var(--lia-bs-gray-900-l) * 0.9))", "secondaryBgColor": "var(--lia-bs-gray-200)", "secondaryBgHoverColor": "hsl(var(--lia-bs-gray-200-h), var(--lia-bs-gray-200-s), calc(var(--lia-bs-gray-200-l) * 0.96))", "secondaryBgActiveColor": "hsl(var(--lia-bs-gray-200-h), var(--lia-bs-gray-200-s), calc(var(--lia-bs-gray-200-l) * 0.92))", "secondaryBorder": "1px solid transparent", "secondaryBorderHover": "1px solid transparent", "secondaryBorderActive": "1px solid transparent", "secondaryBorderFocus": "1px solid transparent", "secondaryBoxShadowFocus": "0 0 1px var(--lia-bs-primary), 0 0 4px hsla(var(--lia-bs-primary-h), var(--lia-bs-primary-s), var(--lia-bs-primary-l), 0.2)", "tertiaryTextColor": "var(--lia-bs-gray-900)", "tertiaryTextHoverColor": "hsl(var(--lia-bs-gray-900-h), var(--lia-bs-gray-900-s), calc(var(--lia-bs-gray-900-l) * 0.95))", "tertiaryTextActiveColor": "hsl(var(--lia-bs-gray-900-h), var(--lia-bs-gray-900-s), calc(var(--lia-bs-gray-900-l) * 0.9))", "tertiaryBgColor": "transparent", "tertiaryBgHoverColor": "transparent", "tertiaryBgActiveColor": "hsla(var(--lia-bs-black-h), var(--lia-bs-black-s), var(--lia-bs-black-l), 0.04)", "tertiaryBorder": "1px solid transparent", "tertiaryBorderHover": "1px solid hsla(var(--lia-bs-black-h), var(--lia-bs-black-s), var(--lia-bs-black-l), 0.08)", "tertiaryBorderActive": "1px solid transparent", "tertiaryBorderFocus": "1px solid transparent", "tertiaryBoxShadowFocus": "0 0 1px var(--lia-bs-primary), 0 0 4px hsla(var(--lia-bs-primary-h), var(--lia-bs-primary-s), var(--lia-bs-primary-l), 0.2)", "destructiveTextColor": "var(--lia-bs-danger)", "destructiveTextHoverColor": "hsl(var(--lia-bs-danger-h), var(--lia-bs-danger-s), calc(var(--lia-bs-danger-l) * 0.95))", "destructiveTextActiveColor": "hsl(var(--lia-bs-danger-h), var(--lia-bs-danger-s), calc(var(--lia-bs-danger-l) * 0.9))", "destructiveBgColor": "var(--lia-bs-gray-200)", "destructiveBgHoverColor": "hsl(var(--lia-bs-gray-200-h), var(--lia-bs-gray-200-s), calc(var(--lia-bs-gray-200-l) * 0.96))", "destructiveBgActiveColor": "hsl(var(--lia-bs-gray-200-h), var(--lia-bs-gray-200-s), calc(var(--lia-bs-gray-200-l) * 0.92))", "destructiveBorder": "1px solid transparent", "destructiveBorderHover": "1px solid transparent", "destructiveBorderActive": "1px solid transparent", "destructiveBorderFocus": "1px solid transparent", "destructiveBoxShadowFocus": "0 0 1px var(--lia-bs-primary), 0 0 4px hsla(var(--lia-bs-primary-h), var(--lia-bs-primary-s), var(--lia-bs-primary-l), 0.2)", "__typename": "ButtonsThemeSettings" }, "border": { "color": "hsla(var(--lia-bs-black-h), var(--lia-bs-black-s), var(--lia-bs-black-l), 0.08)", "mainContent": "NONE", "sideContent": "LIGHT", "radiusSm": "3px", "radius": "5px", "radiusLg": "9px", "radius50": "100vw", "__typename": "BorderT" }, "xs": "0 0 1px hsla(var(--lia-bs-gray-900-h), var(--lia-bs-gray-900-s), var(--lia-bs-gray-900-l), 0.08), 0 3px 0 1px hsla(var(--lia-bs-gray-900-h), var(--lia-bs-gray-900-s), var(--lia-bs-gray-900-l), 0.16)", "sm": "0 2px 4px hsla(var(--lia-bs-gray-900-h), var(--lia-bs-gray-900-s), var(--lia-bs-gray-900-l), 0.12)", "md": "0 5px 15px hsla(var(--lia-bs-gray-900-h), var(--lia-bs-gray-900-s), var(--lia-bs-gray-900-l), 0.3)", "lg": "0 10px 30px hsla(var(--lia-bs-gray-900-h), var(--lia-bs-gray-900-s), var(--lia-bs-gray-900-l), 0.3)", "__typename": "BoxShadowThemeSettings" }, "cards": { "bgColor": "var(--lia-panel-bg-color)", "borderRadius": "var(--lia-panel-border-radius)", "boxShadow": "var(--lia-box-shadow-xs)", "__typename": "CardsThemeSettings" }, "chip": { "maxWidth": "300px", "height": "30px", "__typename": "ChipThemeSettings" }, "coreTypes": { "defaultMessageLinkColor": "var(--lia-bs-link-color)", "defaultMessageLinkDecoration": "none", "defaultMessageLinkFontStyle": "NORMAL", "defaultMessageLinkFontWeight": "400", "defaultMessageLinkFontFamily": "var(--lia-bs-font-family-base)", "forumColor": "#4099E2", "forumFontFamily": "var(--lia-bs-font-family-base)", "forumFontWeight": "var(--lia-default-message-font-weight)", "forumLineHeight": "var(--lia-bs-line-height-base)", "forumFontStyle": "var(--lia-default-message-font-style)", "forumMessageLinkColor": "var(--lia-default-message-link-color)", "forumMessageLinkDecoration": "var(--lia-default-message-link-decoration)", "forumMessageLinkFontStyle": "var(--lia-default-message-link-font-style)", "forumMessageLinkFontWeight": "var(--lia-default-message-link-font-weight)", "forumSolvedColor": "#148563", "blogColor": "#1CBA0", "blogFontFamily": "var(--lia-bs-font-family-base)", "blogFontWeight": "var(--lia-default-message-font-weight)", "blogLineHeight": "1.75", "blogFontStyle": "var(--lia-default-message-font-style)", "blogMessageLinkColor": "var(--lia-default-message-link-color)", "blogMessageLinkDecoration": "var(--lia-default-message-link-decoration)", "blogMessageLinkFontStyle": "var(--lia-default-message-link-font-style)", "blogMessageLinkFontWeight": "var(--lia-default-message-link-font-weight)", "tkbColor": "#4C6B90", "tkbFontFamily": "var(--lia-bs-font-family-base)", "tkbFontWeight": "var(--lia-default-message-font-weight)", "tkbLineHeight": "1.75", "tkbFontStyle": "var(--lia-default-message-font-
```

```
style)","tkbMessageLinkColor":"var(--lia-default-message-link-color)","tkbMessageLinkDecoration":"var(--lia-default-  
message-link-decoration)","tkbMessageLinkFontStyle":"var(--lia-default-message-link-font-  
style)","tkbMessageLinkFontWeight":"var(--lia-default-message-link-font-  
weight)","qandaColor":"#4099E2","qandaFontFamily":"var(--lia-bs-font-family-base)","qandaFontWeight":"var(--lia-  
default-message-font-weight)","qandaLineHeight":"var(--lia-bs-line-height-base)","qandaFontStyle":"var(--lia-default-  
message-link-font-style)","qandaMessageLinkColor":"var(--lia-default-message-link-  
color)","qandaMessageLinkDecoration":"var(--lia-default-message-link-decoration)","qandaMessageLinkFontStyle":"var(--  
lia-default-message-link-font-style)","qandaMessageLinkFontWeight":"var(--lia-default-message-link-font-  
weight)","qandaSolvedColor":"#3FA023","ideaColor":"#FF8000","ideaFontFamily":"var(--lia-bs-font-family-  
base)","ideaFontWeight":"var(--lia-default-message-font-weight)","ideaLineHeight":"var(--lia-bs-line-height-  
base)","ideaFontStyle":"var(--lia-default-message-font-style)","ideaMessageLinkColor":"var(--lia-default-message-link-  
color)","ideaMessageLinkDecoration":"var(--lia-default-message-link-decoration)","ideaMessageLinkFontStyle":"var(--lia-  
default-message-link-font-style)","ideaMessageLinkFontWeight":"var(--lia-default-message-link-font-  
weight)","contestColor":"#FCC845","contestFontFamily":"var(--lia-bs-font-family-base)","contestFontWeight":"var(--lia-  
default-message-font-weight)","contestLineHeight":"var(--lia-bs-line-height-base)","contestFontStyle":"var(--lia-default-  
message-link-font-style)","contestMessageLinkColor":"var(--lia-default-message-link-  
color)","contestMessageLinkDecoration":"var(--lia-default-message-link-  
decoration)","contestMessageLinkFontStyle":"ITALIC","contestMessageLinkFontWeight":"var(--lia-default-message-link-  
font-weight)","occasionColor":"#bc341b","occasionFontFamily":"var(--lia-bs-font-family-  
base)","occasionFontWeight":"var(--lia-default-message-font-weight)","occasionLineHeight":"var(--lia-bs-line-height-  
base)","occasionFontStyle":"var(--lia-default-message-font-style)","occasionMessageLinkColor":"var(--lia-default-  
message-link-color)","occasionMessageLinkDecoration":"var(--lia-default-message-link-  
decoration)","occasionMessageLinkFontStyle":"var(--lia-default-message-link-font-  
style)","occasionMessageLinkFontWeight":"var(--lia-default-message-link-font-  
weight)","grouphubColor":"#333333","categoryColor":"#949494","communityColor":"#FFFFFF","productColor":"#949494","__typename":"CoreTypes"  
{ "black":"#000000","white":"#FFFFFF","gray100":"#F7F7F7","gray200":"#F7F7F7","gray300":"#E8E8E8","gray400":"#D9D9D9","gray500":"#CCCC-  
lia-bs-primary"},"custom":["#D3F5A4","#243A5E"],"__typename":"ColorsThemeSettings"},"divider":  
{ "size":"3px","marginLeft":"4px","marginRight":"4px","borderRadius":"50%","bgColor":"var(--lia-bs-gray-  
600)","bgColorActive":"var(--lia-bs-gray-600)","__typename":"DividerThemeSettings"},"dropdown":{"fontSize":"var(--  
lia-bs-font-size-sm)","borderColor":"var(--lia-bs-border-color)","borderRadius":"var(--lia-bs-border-radius-  
sm)","dividerBg":"var(--lia-bs-gray-300)","itemPaddingY":"5px","itemPaddingX":"20px","headerColor":"var(--lia-bs-gray-  
700)","__typename":"DropdownThemeSettings"},"email":{"link":  
{ "color":"#0069D4","hoverColor":"#0061c2","decoration":"none","hoverDecoration":"underline","__typename":"EmailLinkSettings"},"border":  
{ "color":"#e4e4e4","__typename":"EmailBorderSettings"},"buttons":  
{ "borderRadiusLg":"5px","paddingXLg":"16px","paddingYLG":"7px","fontWeight":"700","primaryTextColor":"#ffffff","primaryTextHoverColor":"#ffffi  
solid transparent"},"primaryBorderHover":"1px solid transparent","__typename":"EmailButtonsSettings"},"panel":  
{ "borderRadius":"5px","borderColor":"#e4e4e4","__typename":"EmailPanelSettings"},"__typename":"EmailThemeSettings"},"emoji":  
{ "skinToneDefault":"#ffcd43","skinToneLight":"#fae3c5","skinToneMediumLight":"#e2cfa5","skinToneMedium":"#daa478","skinToneMediumDark":"#  
{ "color":"var(--lia-bs-body-color)","fontFamily":"Segoe  
UI","fontStyle":"NORMAL","fontWeight":"400","h1FontSize":"34px","h2FontSize":"32px","h3FontSize":"28px","h4FontSize":"24px","h5FontSize":"20  
lia-bs-headings-font-weight)","h2FontWeight":"var(--lia-bs-headings-font-weight)","h3FontWeight":"var(--lia-bs-headings-  
font-weight)","h4FontWeight":"var(--lia-bs-headings-font-weight)","h5FontWeight":"var(--lia-bs-headings-font-  
weight)","h6FontWeight":"var(--lia-bs-headings-font-weight)","__typename":"HeadingThemeSettings"},"icons":  
{ "size10":"10px","size12":"12px","size14":"14px","size16":"16px","size20":"20px","size24":"24px","size30":"30px","size40":"40px","size50":"50px","s  
{ "bgColor":"var(--lia-bs-gray-900)","titleColor":"var(--lia-bs-white)","controlColor":"var(--lia-bs-  
white)","controlBgColor":"var(--lia-bs-gray-800)","__typename":"ImagePreviewThemeSettings"},"input":  
{ "borderColor":"var(--lia-bs-gray-600)","disabledColor":"var(--lia-bs-gray-600)","focusBorderColor":"var(--lia-bs-  
primary)","labelMarginBottom":"10px","btnFontSize":"var(--lia-bs-font-size-sm)","focusBoxShadow":"0 0 3px hsla(var(-  
lia-bs-primary-h), var(--lia-bs-primary-s), var(--lia-bs-primary-l),  
0.2)","checkLabelMarginBottom":"2px","checkboxBorderRadius":"3px","borderRadiusSm":"var(--lia-bs-border-radius-  
sm)","borderRadius":"var(--lia-bs-border-radius)","borderRadiusLg":"var(--lia-bs-border-radius-  
lg)","formTextMarginTop":"4px","textAreaBorderRadius":"var(--lia-bs-border-radius)","activeFillColor":"var(--lia-bs-  
primary)","__typename":"InputThemeSettings"},"loading":{"dotDarkColor":"hsla(var(--lia-bs-black-h), var(--lia-bs-black-  
s), var(--lia-bs-black-l), 0.2)","dotLightColor":"hsla(var(--lia-bs-white-h), var(--lia-bs-white-s), var(--lia-bs-white-l),  
0.5)","barDarkColor":"hsla(var(--lia-bs-black-h), var(--lia-bs-black-s), var(--lia-bs-black-l),  
0.06)","barLightColor":"hsla(var(--lia-bs-white-h), var(--lia-bs-white-s), var(--lia-bs-white-l),  
0.4)","__typename":"LoadingThemeSettings"},"link":{"color":"var(--lia-bs-primary)","hoverColor":"hsl(var(--lia-bs-  
primary-h), var(--lia-bs-primary-s), calc(var(--lia-bs-primary-l) -  
10%))","decoration":"none","hoverDecoration":"underline","__typename":"LinkThemeSettings"},"listGroup":  
{ "itemPaddingY":"15px","itemPaddingX":"15px","borderColor":"var(--lia-bs-gray-  
300)","__typename":"ListGroupThemeSettings"},"modal":{"contentTextColor":"var(--lia-bs-body-  
color)","contentBg":"var(--lia-bs-white)","backgroundBg":"var(--lia-bs-  
black)","smSize":"440px","mdSize":"760px","lgSize":"1080px","backdropOpacity":0.3,"contentBoxShadowXs":"var(--lia-  
bs-box-shadow-sm)","contentBoxShadow":"var(--lia-bs-box-
```

```

shadow)", "headerFontWeight": "700", "__typename": "ModalThemeSettings"}, "navbar": {"position": "FIXED", "background":
{"attachment": null, "clip": null, "color": "var(--lia-bs-
white)", "imageAssetName": "", "imageLastModified": "0", "origin": null, "position": "CENTER_CENTER", "repeat": "NO_REPEAT", "size": "COVER", "__ty
solid var(--lia-bs-border-color)", "boxShadow": "var(--lia-bs-box-shadow-
sm)", "brandMarginRight": "30px", "brandMarginRightSm": "10px", "brandLogoHeight": "30px", "linkGap": "10px", "linkJustifyContent": "flex-
start", "linkPaddingY": "5px", "linkPaddingX": "10px", "linkDropdownPaddingY": "9px", "linkDropdownPaddingX": "var(--lia-
nav-link-px)", "linkColor": "var(--lia-bs-body-color)", "linkHoverColor": "var(--lia-bs-primary)", "linkFontSize": "var(--lia-bs-
font-size-
sm)", "linkFontStyle": "NORMAL", "linkFontWeight": "400", "linkTextTransform": "NONE", "linkLetterSpacing": "normal", "linkBorderRadius": "var(-
lia-bs-border-radius-
sm)", "linkBgColor": "transparent", "linkBgHoverColor": "transparent", "linkBorder": "none", "linkBorderHover": "none", "linkBoxShadow": "none", "linkBox
lia-bs-black-h), var(--lia-bs-black-s), var(--lia-bs-black-l), 0.08)", "controllerBgHoverColor": "hsla(var(--lia-bs-black-h),
var(--lia-bs-black-s), var(--lia-bs-black-l), 0.1)", "controllerIconColor": "var(--lia-bs-body-
color)", "controllerIconHoverColor": "var(--lia-bs-body-color)", "controllerTextColor": "var(--lia-nav-controller-icon-
color)", "controllerTextHoverColor": "var(--lia-nav-controller-icon-hover-color)", "controllerHighlightColor": "hsla(30, 100%,
50%)", "controllerHighlightTextColor": "var(--lia-yiq-light)", "controllerBorderRadius": "var(--lia-border-radius-
50)", "hamburgerColor": "var(--lia-nav-controller-icon-color)", "hamburgerHoverColor": "var(--lia-nav-controller-icon-
color)", "hamburgerBgColor": "transparent", "hamburgerBgHoverColor": "transparent", "hamburgerBorder": "none", "hamburgerBorderHover": "none", "colla
lia-nav-link-color)", "collapseMenuDividerOpacity": 0.16, "__typename": "NavbarThemeSettings"}, "pager":
{"textColor": "var(--lia-bs-link-color)", "textFontWeight": "var(--lia-font-weight-md)", "textFontSize": "var(--lia-bs-font-size-
sm)", "__typename": "PagerThemeSettings"}, "panel": {"bgColor": "var(--lia-bs-white)", "borderRadius": "var(--lia-bs-border-
radius)", "borderColor": "var(--lia-bs-border-color)", "boxShadow": "none", "__typename": "PanelThemeSettings"}, "popover":
{"arrowHeight": "8px", "arrowWidth": "16px", "maxWidth": "300px", "minWidth": "100px", "headerBg": "var(--lia-bs-
white)", "borderColor": "var(--lia-bs-border-color)", "borderRadius": "var(--lia-bs-border-radius)", "boxShadow": "0 0.5rem
1rem hsla(var(--lia-bs-black-h), var(--lia-bs-black-s), var(--lia-bs-black-l),
0.15)", "__typename": "PopoverThemeSettings"}, "prism": {"color": "#000000", "bgColor": "#f5f2f0", "fontFamily": "var(--font-
family-monospace)", "fontSize": "var(--lia-bs-font-size-base)", "fontWeightBold": "var(--lia-bs-font-weight-
bold)", "fontStyleItalic": "italic", "tabSize": 2, "highlightColor": "#b3d4fc", "commentColor": "#62707e", "punctuationColor": "#6f6f6f", "namespaceOpacity": "
0%, 100%,
0.5)", "keywordColor": "#0076a9", "functionColor": "#d3284b", "variableColor": "#c14700", "__typename": "PrismThemeSettings"}, "rte":
{"bgColor": "var(--lia-bs-white)", "borderRadius": "var(--lia-panel-border-radius)", "boxShadow": "var(--lia-panel-box-
shadow)", "customColor1": "#bfd2d2", "customColor2": "#fbee88", "customColor3": "#f8cac6", "customColor4": "#eeca4a", "customColor5": "#c2e0f4", "custo
53%, 51%, 0.4)", "diffChangedColor": "hsla(43, 97%, 63%, 0.4)", "diffNoneColor": "hsla(0, 0%, 80%,
0.4)", "diffRemovedColor": "hsla(9, 74%, 47%,
0.4)", "specialMessageHeaderMarginTop": "40px", "specialMessageHeaderMarginBottom": "20px", "specialMessageItemMarginTop": "0", "specialMessageI
lia-bs-gray-
700)", "tableBorderStyle": "solid", "tableCellPaddingX": "5px", "tableCellPaddingY": "5px", "tableTextColor": "var(--lia-bs-
body-color)", "tableVerticalAlign": "middle", "__typename": "RteThemeSettings"}, "tags": {"bgColor": "var(--lia-bs-gray-
200)", "bgHoverColor": "var(--lia-bs-gray-400)", "borderRadius": "var(--lia-bs-border-radius-sm)", "color": "var(--lia-bs-body-
color)", "hoverColor": "var(--lia-bs-body-color)", "fontWeight": "var(--lia-font-weight-md)", "fontSize": "var(--lia-font-size-
xxs)", "textTransform": "UPPERCASE", "letterSpacing": "0.5px", "__typename": "TagsThemeSettings"}, "toasts":
{"borderRadius": "var(--lia-bs-border-radius)", "paddingX": "12px", "__typename": "ToastsThemeSettings"}, "typography":
{"fontFamilyBase": "Segoe
UI", "fontStyleBase": "NORMAL", "fontWeightBase": "400", "fontWeightLight": "300", "fontWeightNormal": "400", "fontWeightMd": "500", "fontWeightBold
[{"source": "SERVER", "name": "Segoe UI", "styles": [{"style": "NORMAL", "weight": "400", "__typename": "FontStyleData"},
{"style": "NORMAL", "weight": "300", "__typename": "FontStyleData"},
{"style": "NORMAL", "weight": "600", "__typename": "FontStyleData"},
{"style": "NORMAL", "weight": "700", "__typename": "FontStyleData"},
{"style": "ITALIC", "weight": "400", "__typename": "FontStyleData"}], "assetNames": ["SegoeUI-normal-
400.woff2", "SegoeUI-normal-300.woff2", "SegoeUI-normal-600.woff2", "SegoeUI-normal-700.woff2", "SegoeUI-italic-
400.woff2"], "__typename": "CustomFont"}, {"source": "SERVER", "name": "MWF Fluent Icons", "styles":
[{"style": "NORMAL", "weight": "400", "__typename": "FontStyleData"}], "assetNames": ["MWFFluentIcons-normal-
400.woff2"], "__typename": "CustomFont"}, "__typename": "TypographyThemeSettings"}, "unstyledListItem":
{"marginBottomSm": "5px", "marginBottomMd": "10px", "marginBottomLg": "15px", "marginBottomXl": "20px", "marginBottomXxl": "25px", "__typename":
{"light": "#ffffff", "dark": "#000000", "__typename": "YiqThemeSettings"}, "colorLightness":
{"primaryDark": 0.36, "primaryLight": 0.74, "primaryLighter": 0.89, "primaryLightest": 0.95, "infoDark": 0.39, "infoLight": 0.72, "infoLighter": 0.85, "infoLighte
shared/client/components/common/Loading/LoadingDot-1775111750899":
{"__typename": "CachedAsset", "id": "text:en_US-shared/client/components/common/Loading/LoadingDot-
1775111750899", "value":
{"title": "Loading...", "localOverride": false, "CachedAsset": "quilt:o365.prod:pages/blogs/BlogMessagePage:board:MicrosoftSentinelBlog-
1775111749122":
{"__typename": "CachedAsset", "id": "quilt:o365.prod:pages/blogs/BlogMessagePage:board:MicrosoftSentinelBlog-
1775111749122", "value": {"id": "BlogMessagePage", "container": {"id": "Common", "headerProps":
{"backgroundImageProps": null, "backgroundColor": null, "addComponents": null, "removeComponents":

```

```
[{"community.widget.bannerWidget"}, {"componentOrder": null, "__typename": "QuiltContainerSectionProps"}, {"headerComponentProps": {"community.widget.breadcrumbWidget": {"disableLastCrumbForDesktop": false}}, {"footerProps": null, {"footerComponentProps": null, "items": [{"id": "blog-article", "layout": "ONE_COLUMN", "bgColor": null, "showTitle": null, "showDescription": null, "textPosition": null, "textColor": null, "sectionEditLevel": "LOCAL", "main": [{"id": "blogs.widget.blogArticleWidget", "className": "lia-blog-container", "props": null, "__typename": "QuiltComponent"}, {"__typename": "OneSectionColumns"}]}, {"id": "section-1729184836777", "layout": "MAIN_SIDE", "bgColor": "transparent", "showTitle": false, "showDescription": false, "textPosition": "CENTER", "textColor": "var(--lia-bs-body-color)", "sectionEditLevel": null, "bgImage": null, "disableSpacing": null, "edgeToEdgeDisplay": null, "fullHeight": null, "showBorder": null, "__typename": "Ma"}, {"main": [{"side": [{"id": "custom.widget.UnregisteredCTAWidget", "className": null, "props": {"widgetVisibility": "anonymousOnly", "useTitle": true, "useBackground": false, "title": "", "lazyLoad": false, "widgetChooser": "custom.widget.UnregisteredC"}, {"components/common/EmailVerification-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/common/EmailVerification-1775111750899", "value": {"email.verification.title": "Email Verification Required", "email.verification.message.update.email": "To participate in the community, you must first verify your email address. The verification email was sent to {email}. To change your email, visit My Settings.", "email.verification.message.resend.email": "To participate in the community, you must first verify your email address. The verification email was sent to {email}. Resend email."}, {"localOverride": false}], {"CachedAsset: text:en_US-pages/blogs/BlogMessagePage-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-pages/blogs/BlogMessagePage-1775111750899", "value": {"title": "{contextMessageSubject} | {communityTitle}", "errorMissing": "This blog post cannot be found", "name": "Blog Message Page", "section.blog-article.title": "Blog Post", "archivedMessageTitle": "This Content Has Been Archived", "section.section-1729184836777.title": "", "section.section-1729184836777.description": "", "section.CncIde.title": "Blog Post", "section.tifEmD.description": "", "section.tifEmD.title": ""}, {"localOverride": false}], {"CachedAsset: quiltWrapper:o365.prod:Common:1775111735077": {"__typename": "CachedAsset", "id": "quiltWrapper:o365.prod:Common:1775111735077", "value": {"id": "Common", "header": {"backgroundImageProps": {"assetName": null, "backgroundSize": "COVER", "backgroundRepeat": "NO_REPEAT", "backgroundPosition": "CENTER_CENTER", "lastModified": null}, [{"id": "community.widget.navbarWidget", "props": {"showUserName": true, "showRegisterLink": true, "useIconLanguagePicker": true, "useLabelLanguagePicker": true, "style": {"boxShadow": "var(--lia-bs-box-shadow-sm)", "linkFontWeight": "400", "controllerHighlightColor": "hsla(30, 100%, 50%)", "dropdownDividerMarginBottom": "10px", "hamburgerBorderHover": "none", "linkFontSize": "14px", "linkBoxShadowHover": "none", "backgroundC(--lia-border-radius-50)", "hamburgerBgColor": "transparent", "linkTextBorderBottom": "none", "hamburgerColor": "var(--lia-nav-controller-icon-color)", "brandLogoHeight": "30px", "linkLetterSpacing": "normal", "linkBgHoverColor": "transparent", "collapseMenuDividerOpacity": 0.16, "paddingBottom solid var(--lia-bs-border-color)", "hamburgerBorder": "none", "dropdownPaddingX": "10px", "brandMarginRightSm": "10px", "linkBoxShadow": "none", "linkJustifyContent": "flex-start", "linkColor": "var(--lia-bs-body-color)", "collapseMenuDividerBg": "var(--lia-nav-link-color)", "dropdownPaddingTop": "10px", "controllerTextColor": "var(--lia-nav-controller-icon-color)", "controllerHighlightTextColor": "var(--lia-yiq-dark)", "background": {"imageAssetName": "", "color": "var(--lia-bs-white)", "size": "COVER", "repeat": "NO_REPEAT", "position": "CENTER_CENTER", "imageLastModified": ""}, "linkBorderRadius": "var(--lia-bs-border-radius-sm)", "linkHoverColor": "var(--lia-bs-body-color)", "position": "FIXED", "linkBorder": "none", "linkTextBorderBottomHover": "2px solid var(--lia-bs-primary)", "brandMarginRight": "30px", "hamburgerHoverColor": "var(--lia-nav-controller-icon-color)", "linkBorderHover": "none", "collapseMenuMarginLeft": "20px", "linkFontStyle": "NORMAL", "linkPaddingX": "10px", "controllerTextHoverColor": "var(--lia-nav-controller-icon-hover-color)", "paddingTop": "15px", "linkPaddingY": "5px", "linkTextTransform": "NONE", "dropdownBorderColor": "hsla(var(--lia-bs-black-h), var(--lia-bs-black-s), var(--lia-bs-black-l), 0.08)", "controllerBgHoverColor": "hsla(var(--lia-bs-black-h), var(--lia-bs-black-s), var(--lia-bs-black-l), 0.1)", "linkDropdownPaddingX": "var(--lia-nav-link-px)", "linkBgColor": "transparent", "linkDropdownPaddingY": "9px", "controllerIconColor": "var(--lia-bs-body-color)", "dropdownDividerMarginTop": "10px", "linkGap": "10px", "controllerIconHoverColor": "var(--lia-bs-body-color)", "links": {"sideLinks": [], "logoLinks": [], "mainLinks": [{"children": [{"linkType": "INTERNAL", "id": "gxcuf89792", "params": {}, "routeName": "CommunityPage"}, {"children": [{"linkType": "EXTERNAL", "id": "community-hub-link", "url": "/Directory", "target": "SELF"}, {"children": [{"linkType": "INTERNAL", "id": "Common-microsoft365-link", "params": {"categoryId": "microsoft365", "routeName": "CategoryPage"}, {"linkType": "INTERNAL", "id": "Common-windows-link", "params": {"categoryId": "Windows"}, {"routeName": "CategoryPage"}, {"linkType": "INTERNAL", "id": "Common-microsoft-security-link", "params": {"categoryId": "microsoft-security"}, {"routeName": "CategoryPage"}, {"linkType": "INTERNAL", "id": "Common-microsoft-teams-link", "params": {"categoryId": "MicrosoftTeams"}, {"routeName": "CategoryPage"}, {"linkType": "INTERNAL", "id": "Common-azure-link", "params": {"categoryId": "Azure"}, {"routeName": "CategoryPage"}, {"linkType": "INTERNAL", "id": "Common-content_management-link", "params": {"categoryId": "Content_Management"}, {"routeName": "CategoryPage"}, {"linkType": "INTERNAL", "id": "Common-microsoftintune-link", "params": {"categoryId": "microsoftintune"}, {"routeName": "CategoryPage"}, {"linkType": "INTERNAL", "id": "Common-exchange-link", "params": {"categoryId": "Exchange"}, {"routeName": "CategoryPage"}, {"linkType": "INTERNAL", "id": "Common-windows-server-link", "params": {"categoryId": "Windows-Server"}, {"routeName": "CategoryPage"},
```

```
{ "linkType": "INTERNAL", "id": "Common-outlook-link", "params":  
  { "categoryId": "Outlook", "routeName": "CategoryPage", { "linkType": "INTERNAL", "id": "Common-microsoft365-copilot-  
link", "params": { "categoryId": "Microsoft365Copilot", "routeName": "CategoryPage",  
  { "linkType": "EXTERNAL", "id": "Common_Enntvz-view-all-products-  
link", "url": "/Directory", "target": "SELF" }, { "linkType": "EXTERNAL", "id": "products-link", "url": "/", "target": "SELF",  
  { "children": [ { "linkType": "INTERNAL", "id": "Common-education-sector-link", "params":  
    { "categoryId": "EducationSector", "routeName": "CategoryPage", { "linkType": "INTERNAL", "id": "Common-partner-  
community-link", "params": { "categoryId": "PartnerCommunity", "routeName": "CategoryPage",  
  { "linkType": "INTERNAL", "id": "Common-healthcare-and-life-sciences-link", "params":  
    { "categoryId": "HealthcareAndLifeSciences", "routeName": "CategoryPage", { "linkType": "INTERNAL", "id": "Common-i-  
t-ops-talk-link", "params": { "categoryId": "ITOpsTalk", "routeName": "CategoryPage",  
  { "linkType": "INTERNAL", "id": "Common-public-sector-link", "params":  
    { "categoryId": "PublicSector", "routeName": "CategoryPage", { "linkType": "INTERNAL", "id": "Common-microsoftfor-  
nonprofits-link", "params": { "categoryId": "MicrosoftforNonprofits", "routeName": "CategoryPage",  
  { "linkType": "INTERNAL", "id": "Common-io-t-link", "params": { "categoryId": "IoT", "routeName": "CategoryPage",  
  { "linkType": "INTERNAL", "id": "Common-mvp-link", "params": { "categoryId": "mvp", "routeName": "CategoryPage",  
  { "linkType": "INTERNAL", "id": "Common-microsoft-mechanics-link", "params":  
    { "categoryId": "MicrosoftMechanics", "routeName": "CategoryPage", { "linkType": "INTERNAL", "id": "Common-driving-  
adoption-link", "params": { "categoryId": "DrivingAdoption", "routeName": "CategoryPage",  
  { "linkType": "INTERNAL", "id": "Common-microsoft-learn-for-educators-link", "params": { "categoryId": "microsoft-learn-  
for-educators", "routeName": "CategoryPage", { "linkType": "EXTERNAL", "id": "topics-link", "url": "/", "target": "SELF",  
  { "children": [ { "linkType": "EXTERNAL", "id": "all-blogs-link", "url": "/Blogs", "target": "SELF", { "children":  
    [ { "linkType": "EXTERNAL", "id": "all-events-link", "url": "/Events", "target": "SELF", { "children":  
      [ { "linkType": "INTERNAL", "id": "Skills-Hub-link", "params": { "categoryId": "skills-hub", "routeName": "CategoryPage",  
        { "linkType": "INTERNAL", "id": "Skills-Hub-Blog", "params": { "boardId": "skills-hub-blog", "categoryId": "skills-  
hub", "routeName": "BlogBoardPage", { "linkType": "EXTERNAL", "id": "ms-learn-ext-LD", "url": "/category/skills-hub?  
tab=group", "target": "BLANK", { "linkType": "EXTERNAL", "id": "ms-learn-ext-  
dynamics", "url": "https://docs.microsoft.com/learn/dynamics365/?WT.mc_id=techcom_header-webpage-  
m365", "target": "BLANK", { "linkType": "EXTERNAL", "id": "ms-learn-ext-  
m365", "url": "https://docs.microsoft.com/learn/m365/?wt.mc_id=techcom_header-webpage-m365", "target": "BLANK",  
  { "linkType": "EXTERNAL", "id": "ms-learn-ext-security", "url": "https://docs.microsoft.com/learn/topics/sci/?  
wt.mc_id=techcom_header-webpage-m365", "target": "BLANK", { "linkType": "EXTERNAL", "id": "ms-learn-ext-  
pp", "url": "https://docs.microsoft.com/learn/powerplatform/?wt.mc_id=techcom_header-webpage-  
powerplatform", "target": "BLANK", { "linkType": "EXTERNAL", "id": "ms-learn-ext-  
github", "url": "https://docs.microsoft.com/learn/github/?wt.mc_id=techcom_header-webpage-github", "target": "BLANK",  
  { "linkType": "EXTERNAL", "id": "ms-learn-ext-teams", "url": "https://docs.microsoft.com/learn/teams/?  
wt.mc_id=techcom_header-webpage-teams", "target": "BLANK", { "linkType": "EXTERNAL", "id": "ms-learn-ext-  
net", "url": "https://docs.microsoft.com/learn/dotnet/?wt.mc_id=techcom_header-webpage-dotnet", "target": "BLANK",  
  { "linkType": "EXTERNAL", "id": "ms-learn-ext-azure", "url": "https://docs.microsoft.com/learn/azure/?  
WT.mc_id=techcom_header-webpage-m365", "target": "BLANK", { "linkType": "INTERNAL", "id": "Skills-Hub", "params":  
  { "categoryId": "skills-hub", "routeName": "CategoryPage", { "children": [ { "linkType": "INTERNAL", "id": "Common-  
community-info-center-link", "params": { "categoryId": "Community-Info-Center", "routeName": "CategoryPage",  
  { "linkType": "INTERNAL", "id": "Common-usergroups-link", "params":  
  { "categoryId": "usergroups", "routeName": "CategoryPage", { "linkType": "INTERNAL", "id": "Common-community-news-  
desk-link", "params": { "categoryId": "CommunityNewsDesk", "routeName": "CategoryPage",  
  { "linkType": "INTERNAL", "id": "Common-microsoft-global-community-initiative-link", "params": { "categoryId": "microsoft-  
global-community-initiative", "routeName": "CategoryPage", { "linkType": "INTERNAL", "id": "Common-gxcuf89792-  
community", "params":  
  { "routeName": "CommunityPage", "showSearchIcon": true, "languagePickerStyle": "iconAndLabel", " __typename": "QuiltComponent",  
  { "id": "community.widget.breadcrumbWidget", "props": { "backgroundColor": "transparent", "linkHighlightColor": "var(--lia-  
bs-primary)", "visualEffects": { "showBottomBorder": true, "linkTextColor": "var(--lia-bs-gray-  
700)", " __typename": "QuiltComponent", { "id": "custom.widget.CommunityBanner", "props":  
  { "widgetVisibility": "signedInOrAnonymous", "useTitle": true, "usePageWidth": false, "useBackground": false, "title": "", "lazyLoad": false, " __typename": "QuiltComponent",  
  { "id": "custom.widget.ChatbotWidget", "props":  
  { "customComponentId": "custom.widget.ChatbotWidget", "cDisplay_form": true, "useBackground": false, " __typename": "QuiltComponent",  
  { "id": "custom.widget.HeroBanner", "props":  
  { "widgetVisibility": "signedInOrAnonymous", "usePageWidth": false, "useTitle": true, "cMax_items": 3, "useBackground": false, "title": "", "lazyLoad": false, "w  
  { "backgroundImageProps":  
  { "assetName": null, "backgroundSize": "COVER", "backgroundRepeat": "NO_REPEAT", "backgroundPosition": "CENTER_CENTER", "lastModified": null,  
  [ { "id": "custom.widget.SocialSharing", "props":  
  { "widgetVisibility": "signedInOrAnonymous", "useTitle": true, "useBackground": false, "title": "", "lazyLoad": false, " __typename": "QuiltComponent",  
  { "id": "custom.widget.MicrosoftFooter", "props":  
  { "widgetVisibility": "signedInOrAnonymous", "useTitle": true, "useBackground": false, "title": "", "lazyLoad": false, " __typename": "QuiltComponent", " __ty  
components/common/ActionFeedback-1775111750899: { " __typename": "CachedAsset", "id": "text.en_US-
```

components/common/ActionFeedback-1775111750899", "value":
{"joinedGroupHub.title": "Welcome", "joinedGroupHub.message": "You are now a member of this group and are subscribed to updates.", "groupHubInviteNotFound.title": "Invitation Not Found", "groupHubInviteNotFound.message": "Sorry, we could not find your invitation to the group. The owner may have canceled the invite.", "groupHubNotFound.title": "Group Not Found", "groupHubNotFound.message": "The grouphub you tried to join does not exist. It may have been deleted.", "existingGroupHubMember.title": "Already Joined", "existingGroupHubMember.message": "You are already a member of this group.", "accountLocked.title": "Account Locked", "accountLocked.message": "Your account has been locked due to multiple failed attempts. Try again in {lockoutTime} minutes.", "editedGroupHub.title": "Changes Saved", "editedGroupHub.message": "Your group has been updated.", "leftGroupHub.title": "Goodbye", "leftGroupHub.message": "You are no longer a member of this group and will not receive future updates.", "deletedGroupHub.title": "Deleted", "deletedGroupHub.message": "The group has been deleted.", "groupHubCreated.title": "Group Created", "groupHubCreated.message": "{groupHubName} is ready to use", "accountClosed.title": "Account Closed", "accountClosed.message": "The account has been closed and you will now be redirected to the homepage", "resetTokenExpired.title": "Reset Password Link has Expired", "resetTokenExpired.message": "Try resetting your password again", "invalidUrl.title": "Invalid URL", "invalidUrl.message": "The URL you're using is not recognized. Verify your URL and try again.", "accountClosedForUser.title": "Account Closed", "accountClosedForUser.message": "{userName}'s account is closed", "inviteTokenInvalid.title": "Invitation Invalid", "inviteTokenInvalid.message": "Your invitation to the community has been canceled or expired.", "inviteTokenError.title": "Invitation Verification Failed", "inviteTokenError.message": "The url you are utilizing is not recognized. Verify your URL and try again", "pageNotFound.title": "Access Denied", "pageNotFound.message": "You do not have access to this area of the community or it doesn't exist", "eventAttending.title": "Responded as Attending", "eventAttending.message": "You'll be notified when there's new activity and reminded as the event approaches", "eventInterested.title": "Responded as Interested", "eventInterested.message": "You'll be notified when there's new activity and reminded as the event approaches", "eventNotFound.title": "Event Not Found", "eventNotFound.message": "The event you tried to respond to does not exist.", "redirectToRelatedPage.title": "Showing Related Content", "redirectToRelatedPageForBaseUsers.title": "Showing Related Content", "redirectToRelatedPageForBaseUsers.message": "The content you are trying to access is archived", "redirectToRelatedPage.message": "The content you are trying to access is archived", "relatedUrl.archivalLink.flyoutMessage": "The content you are trying to access is archived View Archived Content"}, {"localOverride": false}, {"CachedAsset": {"component": "custom.widget.CommunityBanner-en-us-1775107888865": {"__typename": "CachedAsset", "id": "component:custom.widget.CommunityBanner-en-us-1775107888865", "value": {"component": {"id": "custom.widget.CommunityBanner", "template": {"id": "CommunityBanner", "markupLanguage": "REACT", "style": null, "texts": null, "defaults": {"config": {"applicablePages": [], "description": null, "fetchedContent": null, __typename": "ComponentConfiguration"}, "props": [], __typename": "ComponentProperties"}, "components": [{"id": "custom.widget.CommunityBanner", "form": null, "config": null, "props": [], __typename": "Component"}], "grouping": "CUSTOM", __typename": "ComponentTemplate"}, "properties": {"config": {"applicablePages": [], "description": null, "fetchedContent": null, __typename": "ComponentConfiguration"}, "props": [], __typename": "ComponentProperties"}, "form": null, __typename": "Component", "localOverride": false}, "globalCss": null, "form": null, "localOverride": en-us-1775107888865": {"__typename": "CachedAsset", "id": "component:custom.widget.ChatbotWidget-en-us-1775107888865", "value": {"component": {"id": "custom.widget.ChatbotWidget", "template": {"id": "ChatbotWidget", "markupLanguage": "REACT", "style": null, "texts": {"chatbot.references.title": "Related Articles", "chatbot.welcome.title": "Welcome!", "chatbot.welcome.description": "I'm here to help you explore and discover great content.", "chatbot.welcome.prompt": "Ask me a question or choose a suggestion below to get started:", "chatbot.welcome.cta": "Let's dive in\u2014what would you like to discover today?", "chatbot.status.typing": "Assistant is typing...", "chatbot.status.error": "error", "chatbot.error.response": "Failed to get response. Please try again.", "chatbot.error.processing": "There was an error processing your message.", "chatbot.error.configuration": "API URL not configured", "chatbot.error.network": "Network error occurred. Please check your connection and try again.", "chatbot.error.timeout": "Request timed out. Please try again.", "chatbot.error.emptyResponse": "I couldn't generate a response. Please try rephrasing your question.", "chatbot.buttons.send": "Send", "chatbot.buttons.close": "Close chat", "chatbot.buttons.newChat": "Start new chat", "chatbot.buttons.collapse": "Collapse chat panel", "chatbot.buttons.expand": "Expand chat panel", "chatbot.buttons.fullscreen": "Enter fullscreen", "chatbot.buttons.exitFullscreen": "Exit fullscreen", "chatbot.buttons.like": "Like this response", "chatbot.buttons.dislike": "Dislike this response", "chatbot.buttons.removeLike": "Remove like", "chatbot.buttons.removeDislike": "Remove dislike", "chatbot.aria.chatInput": "Chat input", "chatbot.aria.sendMessage": "Send message", "chatbot.aria.openChat": "Open chat assistant", "chatbot.aria.closeChat": "Close chat assistant", "chatbot.defaults.title": "Ask Tech Community", "chatbot.defaults.subtitle": "Ask questions \u2013 get answers", "chatbot.defaults.entryHeading": "Find answers", "chatbot.defaults.entrySubtext": "Ask the agent", "chatbot.defaults.placeholder": "Type your message...", "chatbot.defaults.initialMessage": "Hi! I'm your assistant. Ask me something or pick a suggestion above to begin.", "chatbot.suggestions.findBlogs": "Find insightful blogs", "chatbot.suggestions.exploreEvents": "Explore upcoming events", "chatbot.suggestions.startJourney": "Start your journey with something new", "chatbot.dialog.endConversation": "End conversation", "chatbot.dialog.confirmEndConversation": "Do you want to end this conversation and start over?", "chatbot.dialog.endConversationButton": "End

```
conversation", "chatbot.dialog.cancel": "Cancel", "chatbot.error.genericServiceUnavailable": "The service is currently
unavailable. Please try again later.", "chatbot.error.noResults": "We could not find any information related to your query. Try
rephrasing your query.", "defaults": {"config": {"applicablePages":
[], "description": null, "fetchedContent": null, "__typename": "ComponentConfiguration"}, "props":
[], "__typename": "ComponentProperties"}, "components":
[{"id": "custom.widget.ChatbotWidget", "form": null, "config": null, "props":
[], "__typename": "ComponentTemplate"}, {"properties": {"config":
{"applicablePages": [], "description": null, "fetchedContent": null, "__typename": "ComponentConfiguration"}, "props":
[], "__typename": "ComponentProperties"}, "form": null, "__typename": "Component", "localOverride": false}, {"globalCss": null, "form": null, "localOverride":
en-us-1775107888865": {"__typename": "CachedAsset", "id": "component:custom.widget.HeroBanner-en-us-
1775107888865", "value": {"component": {"id": "custom.widget.HeroBanner", "template":
{"id": "HeroBanner", "markupLanguage": "REACT", "style": null, "texts": {"searchPlaceholderText": "Search this
community", "followActionText": "Follow", "unfollowActionText": "Following", "searchOnHoverText": "Please enter your
search term(s) and then press return key to complete a search.", "blogs.sidebar.pagetitle": "Latest Blogs | Microsoft Tech
Community", "followThisNode": "Follow this node", "unfollowThisNode": "Unfollow this
node", "customField.teamsLink.title": "Microsoft teams link", "customField.teamsLink.label": "Teams meeting
url"}, "defaults": {"config": {"applicablePages":
[], "description": null, "fetchedContent": null, "__typename": "ComponentConfiguration"}, "props":
[{"id": "max_items", "dataType": "NUMBER", "list": false, "defaultValue": "3", "label": "Max Items", "description": "The
maximum number of items to display in the
carousel", "possibleValues": null, "control": "INPUT", "__typename": "PropDefinition"}, {"__typename": "ComponentProperties"}, "components":
[{"id": "custom.widget.HeroBanner", "form": {"fields":
[{"id": "widgetChooser", "validation": null, "noValidation": null, "dataType": "STRING", "list": null, "control": null, "defaultValue": null, "label": null, "description":
{"id": "title", "validation": null, "noValidation": null, "dataType": "STRING", "list": null, "control": null, "defaultValue": null, "label": null, "description": null, "possi
{"id": "useTitle", "validation": null, "noValidation": null, "dataType": "BOOLEAN", "list": null, "control": null, "defaultValue": null, "label": null, "description": nul
{"id": "useBackground", "validation": null, "noValidation": null, "dataType": "BOOLEAN", "list": null, "control": null, "defaultValue": null, "label": null, "descripti
{"id": "widgetVisibility", "validation": null, "noValidation": null, "dataType": "STRING", "list": null, "control": null, "defaultValue": null, "label": null, "description":
{"id": "moreOptions", "validation": null, "noValidation": null, "dataType": "STRING", "list": null, "control": null, "defaultValue": null, "label": null, "description": n
{"id": "cMax_items", "validation": null, "noValidation": null, "dataType": "NUMBER", "list": false, "control": "INPUT", "defaultValue": "3", "label": "Max
Items", "description": "The maximum number of items to display in the
carousel", "possibleValues": null, "__typename": "FormField"}], "layout": {"rows":
[{"id": "widgetChooserGroup", "type": "fieldset", "as": null, "items":
[{"id": "widgetChooser", "className": null, "__typename": "FormFieldRef"}], "props": null, "legend": null, "description": null, "className": null, "viewVariant":
{"id": "titleGroup", "type": "fieldset", "as": null, "items": [{"id": "title", "className": null, "__typename": "FormFieldRef"}],
{"id": "useTitle", "className": null, "__typename": "FormFieldRef"}], "props": null, "legend": null, "description": null, "className": null, "viewVariant": null, "to
{"id": "useBackground", "type": "fieldset", "as": null, "items":
[{"id": "useBackground", "className": null, "__typename": "FormFieldRef"}], "props": null, "legend": null, "description": null, "className": null, "viewVariant":
{"id": "widgetVisibility", "type": "fieldset", "as": null, "items":
[{"id": "widgetVisibility", "className": null, "__typename": "FormFieldRef"}], "props": null, "legend": null, "description": null, "className": null, "viewVariant":
{"id": "moreOptionsGroup", "type": "fieldset", "as": null, "items":
[{"id": "moreOptions", "className": null, "__typename": "FormFieldRef"}], "props": null, "legend": null, "description": null, "className": null, "viewVariant": nu
{"id": "componentPropsGroup", "type": "fieldset", "as": null, "items":
[{"id": "cMax_items", "className": null, "__typename": "FormFieldRef"}], "props": null, "legend": null, "description": null, "className": null, "viewVariant": null
[], "__typename": "Component"}, {"grouping": "CUSTOM", "__typename": "ComponentTemplate"}, {"properties": {"config":
{"applicablePages": [], "description": null, "fetchedContent": null, "__typename": "ComponentConfiguration"}, "props":
[{"id": "max_items", "dataType": "NUMBER", "list": false, "defaultValue": "3", "label": "Max Items", "description": "The
maximum number of items to display in the
carousel", "possibleValues": null, "control": "INPUT", "__typename": "PropDefinition"}, {"__typename": "ComponentProperties"}, "form":
{"fields":
[{"id": "widgetChooser", "validation": null, "noValidation": null, "dataType": "STRING", "list": null, "control": null, "defaultValue": null, "label": null, "description":
{"id": "title", "validation": null, "noValidation": null, "dataType": "STRING", "list": null, "control": null, "defaultValue": null, "label": null, "description": null, "possi
{"id": "useTitle", "validation": null, "noValidation": null, "dataType": "BOOLEAN", "list": null, "control": null, "defaultValue": null, "label": null, "description": nul
{"id": "useBackground", "validation": null, "noValidation": null, "dataType": "BOOLEAN", "list": null, "control": null, "defaultValue": null, "label": null, "descripti
{"id": "widgetVisibility", "validation": null, "noValidation": null, "dataType": "STRING", "list": null, "control": null, "defaultValue": null, "label": null, "description":
{"id": "moreOptions", "validation": null, "noValidation": null, "dataType": "STRING", "list": null, "control": null, "defaultValue": null, "label": null, "description": n
{"id": "cMax_items", "validation": null, "noValidation": null, "dataType": "NUMBER", "list": false, "control": "INPUT", "defaultValue": "3", "label": "Max
Items", "description": "The maximum number of items to display in the
carousel", "possibleValues": null, "__typename": "FormField"}], "layout": {"rows":
[{"id": "widgetChooserGroup", "type": "fieldset", "as": null, "items":
[{"id": "widgetChooser", "className": null, "__typename": "FormFieldRef"}], "props": null, "legend": null, "description": null, "className": null, "viewVariant":
{"id": "titleGroup", "type": "fieldset", "as": null, "items": [{"id": "title", "className": null, "__typename": "FormFieldRef"}],
{"id": "useTitle", "className": null, "__typename": "FormFieldRef"}], "props": null, "legend": null, "description": null, "className": null, "viewVariant": null, "to
{"id": "useBackground", "type": "fieldset", "as": null, "items":
[{"id": "useBackground", "className": null, "__typename": "FormFieldRef"}], "props": null, "legend": null, "description": null, "className": null, "viewVariant":
```

```

{"id":"widgetVisibility","type":"fieldset","as":null,"items":
[{"id":"widgetVisibility","className":null,"__typename":"FormFieldRef"},"props":null,"legend":null,"description":null,"className":null,"viewVariant":
{"id":"moreOptionsGroup","type":"fieldset","as":null,"items":
[{"id":"moreOptions","className":null,"__typename":"FormFieldRef"},"props":null,"legend":null,"description":null,"className":null,"viewVariant":nu
{"id":"componentPropsGroup","type":"fieldset","as":null,"items":
[{"id":"cMax_items","className":null,"__typename":"FormFieldRef"},"props":null,"legend":null,"description":null,"className":null,"viewVariant":nu
{"fields":
[{"id":"widgetChooser","validation":null,"noValidation":null,"dataType":"STRING","list":null,"control":null,"defaultValue":null,"label":null,"description
{"id":"title","validation":null,"noValidation":null,"dataType":"STRING","list":null,"control":null,"defaultValue":null,"label":null,"description":null,"possi
{"id":"useTitle","validation":null,"noValidation":null,"dataType":"BOOLEAN","list":null,"control":null,"defaultValue":null,"label":null,"description":nul
{"id":"useBackground","validation":null,"noValidation":null,"dataType":"BOOLEAN","list":null,"control":null,"defaultValue":null,"label":null,"descripti
{"id":"widgetVisibility","validation":null,"noValidation":null,"dataType":"STRING","list":null,"control":null,"defaultValue":null,"label":null,"description
{"id":"moreOptions","validation":null,"noValidation":null,"dataType":"STRING","list":null,"control":null,"defaultValue":null,"label":null,"description":n
{"id":"cMax_items","validation":null,"noValidation":null,"dataType":"NUMBER","list":false,"control":"INPUT","defaultValue":"3","label":"Max
Items","description":"The maximum number of items to display in the
carousel","possibleValues":null,"__typename":"FormField"},"layout":{"rows":
[{"id":"widgetChooserGroup","type":"fieldset","as":null,"items":
[{"id":"widgetChooser","className":null,"__typename":"FormFieldRef"},"props":null,"legend":null,"description":null,"className":null,"viewVariant":
{"id":"titleGroup","type":"fieldset","as":null,"items":[{"id":"title","className":null,"__typename":"FormFieldRef"}],
{"id":"useTitle","className":null,"__typename":"FormFieldRef"},"props":null,"legend":null,"description":null,"className":null,"viewVariant":null,"to
{"id":"useBackground","type":"fieldset","as":null,"items":
[{"id":"useBackground","className":null,"__typename":"FormFieldRef"},"props":null,"legend":null,"description":null,"className":null,"viewVariant"
{"id":"widgetVisibility","type":"fieldset","as":null,"items":
[{"id":"widgetVisibility","className":null,"__typename":"FormFieldRef"},"props":null,"legend":null,"description":null,"className":null,"viewVariant"
{"id":"moreOptionsGroup","type":"fieldset","as":null,"items":
[{"id":"moreOptions","className":null,"__typename":"FormFieldRef"},"props":null,"legend":null,"description":null,"className":null,"viewVariant":nu
{"id":"componentPropsGroup","type":"fieldset","as":null,"items":
[{"id":"cMax_items","className":null,"__typename":"FormFieldRef"},"props":null,"legend":null,"description":null,"className":null,"viewVariant":nu
en-us-1775107888865":{"__typename":"CachedAsset","id":"component:custom.widget.UnregisteredCTAWidget-en-us-
1775107888865","value":{"component":{"id":"custom.widget.UnregisteredCTAWidget","template":
{"id":"UnregisteredCTAWidget","markupLanguage":"REACT","style":null,"texts":{"register.communityHub":"Welcome to
the {name} Community Hub. Sign in to like, participate, or start a conversation.",register.category:"Welcome to the
{name} Community Hub. Sign in to like, participate, or start a conversation.",register.discussionBoard:"Welcome to the
{name} space. Sign in to like, reply, or start a discussion.",register.blogSpace:"Welcome to the {name} space. Sign in to
like or comment on articles in this space.",register.eventSpace:"Welcome to the {name} space. Sign in to RSVP, add
events to your calendar, and join the conversation.",register.ideaSpace:"Welcome to the {name} space. Sign in to vote,
comment, or submit your own feedback.",buttonRegister:"Sign in",register.discussionBoardArticle:"Have a question or
insight to share? Sign in to join the discussion.",register.blogSpaceArticle:"Enjoying the article? Sign in to share your
thoughts.",register.eventSpaceArticle:"Don't just watch - take part. Sign in to RSVP, ask questions, and join the
discussion.",register.ideaSpaceArticle:"Sign in to submit ideas, upvote ideas, and join the conversation."},"defaults":
{"config":{"applicablePages":
[],"description":null,"fetchedContent":null,"__typename":"ComponentConfiguration"},"props":
[],"__typename":"ComponentProperties"},"components":
[{"id":"custom.widget.UnregisteredCTAWidget","form":null,"config":null,"props":
[],"__typename":"Component"},"grouping":"CUSTOM","__typename":"ComponentTemplate"},"properties":{"config":
{"applicablePages":[],"description":null,"fetchedContent":null,"__typename":"ComponentConfiguration"},"props":
[],"__typename":"ComponentProperties"},"form":null,"__typename":"Component","localOverride":false},"globalCss":null,"form":null},"localOverride":
en-us-1775107888865":{"__typename":"CachedAsset","id":"component:custom.widget.SocialSharing-en-us-
1775107888865","value":{"component":{"id":"custom.widget.SocialSharing","template":
{"id":"SocialSharing","markupLanguage":"HANDLEBARS","style":"sharePage {\n display: flex;\n justify-content:
center;\n background: #d7d7d7;\n padding: 0px;\n height: 60px;\n}\n.singleSocialIcons {\n display: flex;\n gap: 12px;\n list-
style-type: none;\n padding: 0px;\n margin: 0;\n}\n.containers {\n display: flex;\n gap: 30px;\n}\n\n.listIcon {\n align-
content: center;\n}\n.headingShare {\n display: inline;\n margin-right: 25px;\n margin-bottom: 0px;\n font-size: 20px;\n
font-weight: 550;\n align-content: center;\n}\n\n@media (max-width: 990px) {\n .sharePage {\n display: flex;\n justify-
content: center;\n }\n\n .containers {\n display: inline-block;\n justify-content: center;\n align-content: center;\n align-items:
center;\n }\n\n .headingShare {\n display: flex;\n justify-content: center;\n }\n\n .singleSocialIcons {\n
}\n\n"},"texts":null,"defaults":{"config":{"applicablePages":[],"description":"Adds buttons to share to various social media
websites","fetchedContent":null,"__typename":"ComponentConfiguration"},"props":
[],"__typename":"ComponentProperties"},"components":
[{"id":"custom.widget.SocialSharing","form":null,"config":null,"props":
[],"__typename":"Component"},"grouping":"CUSTOM","__typename":"ComponentTemplate"},"properties":{"config":
{"applicablePages":[],"description":"Adds buttons to share to various social media
websites","fetchedContent":null,"__typename":"ComponentConfiguration"},"props":
[],"__typename":"ComponentProperties"},"form":null,"__typename":"Component","localOverride":false},"globalCss":

```

```
 {"css": ".custom_widget_SocialSharing_sharePage_6x3n8_1 {\n display: flex;\n justify-content: center;\n background: #d7d7d7;\n padding: 0;\n height: 3.75rem;\n}\n.custom_widget_SocialSharing_singleSocialIcons_6x3n8_8 {\n display: flex;\n gap: 0.75rem;\n list-style-type: none;\n padding: 0;\n margin: 0;\n}\n.custom_widget_SocialSharing_containers_6x3n8_15 {\n display: flex;\n gap: 1.875rem;\n}\n.custom_widget_SocialSharing_listIcon_6x3n8_20 {\n align-content: center;\n}\n.custom_widget_SocialSharing_headingShare_6x3n8_23 {\n display: inline;\n margin-right: 1.5625rem;\n margin-bottom: 0;\n font-size: 1.25rem;\n font-weight: 550;\n align-content: center;\n}\n@media (max-width: 990px) {\n .custom_widget_SocialSharing_sharePage_6x3n8_1 {\n display: flex;\n justify-content: center;\n }\n}\n .custom_widget_SocialSharing_containers_6x3n8_15 {\n display: inline-block;\n justify-content: center;\n align-content: center;\n align-items: center;\n }\n .custom_widget_SocialSharing_headingShare_6x3n8_23 {\n display: flex;\n justify-content: center;\n }\n .custom_widget_SocialSharing_singleSocialIcons_6x3n8_8 {\n }\n}\n", "tokens": {"sharePage": ".custom_widget_SocialSharing_sharePage_6x3n8_1", "singleSocialIcons": ".custom_widget_SocialSharing_singleSocialIcons_6x3n8_8", "en-us-1775107888865": {"__typename": "CachedAsset", "id": "component:custom.widget.MicrosoftFooter-en-us-1775107888865", "value": {"component": {"id": "custom.widget.MicrosoftFooter", "template": {"id": "MicrosoftFooter", "markupLanguage": "HANDLEBARS", "style": ".context-uhf {\n\n min-width: 280px;\n\n font-size: 15px;\n\n box-sizing: border-box;\n\n -ms-text-size-adjust: 100%;\n\n -webkit-text-size-adjust: 100%;\n\n & * & * & *:before, \n\n & *:after {\n\n box-sizing: inherit;\n\n }\n\n a.c-uhff-link {\n\n color: #616161;\n\n word-break: break-word;\n\n text-decoration: none;\n\n }\n\n &a:link, \n\n &a:focus, \n\n &a:hover, \n\n &a:active, \n\n &a:visited {\n\n text-decoration: none;\n\n color: inherit;\n\n }\n\n & div {\n\n font-family: 'Segoe UI', SegoeUI, 'Helvetica Neue', Helvetica, Arial, sans-serif;\n\n }\n\n .c-uhff {\n\n background: #f2f2f2;\n\n margin: -1.5625;\n\n width: auto;\n\n height: auto;\n\n }\n\n .c-uhff-nav {\n\n margin: 0 auto;\n\n max-width: calc(1600px + 10%);\n\n padding: 0 5%;\n\n box-sizing: inherit;\n\n &:before, \n\n &:after {\n\n content: ' '\n\n display: table;\n\n clear: left;\n\n }\n\n @media only screen and (max-width: 1083px) {\n\n padding-left: 12px;\n\n }\n\n .c-heading-4 {\n\n color: #616161;\n\n word-break: break-word;\n\n font-size: 15px;\n\n line-height: 20px;\n\n padding: 36px 0 4px;\n\n font-weight: 600;\n\n }\n\n .c-uhff-nav-row {\n\n .c-uhff-nav-group {\n\n display: block;\n\n float: left;\n\n min-height: 1px;\n\n vertical-align: text-top;\n\n padding: 0 12px;\n\n width: 100%;\n\n zoom: 1;\n\n &:first-child {\n\n padding-left: 0;\n\n @media only screen and (max-width: 1083px) {\n\n padding-left: 12px;\n\n }\n\n }\n\n @media only screen and (min-width: 540px) and (max-width: 1082px) {\n\n width: 33.33333%;\n\n }\n\n @media only screen and (min-width: 1083px) {\n\n width: 16.6666666667%;\n\n }\n\n ul.c-list.f-bare {\n\n font-size: 11px;\n\n line-height: 16px;\n\n margin-top: 0;\n\n margin-bottom: 0;\n\n padding-left: 0;\n\n list-style-type: none;\n\n li {\n\n word-break: break-word;\n\n padding: 8px 0;\n\n margin: 0;\n\n }\n\n }\n\n .c-uhff-base {\n\n background: #f2f2f2;\n\n margin: 0 auto;\n\n max-width: calc(1600px + 10%);\n\n padding: 30px 5% 16px;\n\n &:before, \n\n &:after {\n\n content: ' '\n\n display: table;\n\n }\n\n &:after {\n\n clear: both;\n\n }\n\n a.c-uhff-ccpa, \n\n a.c-uhff-consumer {\n\n display: flex;\n\n float: left;\n\n font-size: 11px;\n\n line-height: 16px;\n\n padding: 4px 24px 0 0;\n\n }\n\n a.c-uhff-ccpa:hover, \n\n a.c-uhff-consumer:hover {\n\n text-decoration: underline;\n\n }\n\n ul.c-list {\n\n font-size: 11px;\n\n line-height: 16px;\n\n float: right;\n\n margin: 3px 0;\n\n color: #616161;\n\n li {\n\n padding: 0 24px 4px 0;\n\n display: inline-block;\n\n }\n\n .c-list.f-bare {\n\n padding-left: 0;\n\n list-style-type: none;\n\n }\n\n @media only screen and (max-width: 1083px) {\n\n display: flex;\n\n flex-wrap: wrap;\n\n padding: 30px 24px 16px;\n\n }\n\n .social-share {\n\n position: fixed;\n\n top: 60%;\n\n transform: translateY(-50%);\n\n left: 0;\n\n z-index: 1000;\n\n }\n\n .sharing-options {\n\n list-style: none;\n\n padding: 0;\n\n margin: 0;\n\n display: block;\n\n flex-direction: column;\n\n background-color: white;\n\n width: 50px;\n\n border-radius: 0px 7px 7px 0px;\n\n }\n\n .linkedin-icon {\n\n border-top-right-radius: 7px;\n\n }\n\n .linkedin-icon:hover {\n\n border-radius: 0;\n\n }\n\n .social-share-email-image:hover {\n\n border-radius: 0;\n\n }\n\n .social-link-footer:hover .linkedin-icon {\n\n border-radius: 0;\n\n }\n\n .social-link-footer:hover .social-share-email-image {\n\n border-radius: 0;\n\n }\n\n .social-link-footer img {\n\n width: 30px;\n\n height: auto;\n\n transition: filter 0.3s ease;\n\n }\n\n .social-share-list {\n\n width: 50px;\n\n }\n\n .social-share-rss-image {\n\n width: 30px;\n\n height: auto;\n\n transition: filter 0.3s ease;\n\n }\n\n .sharing-options li {\n\n width: 50px;\n\n height: 50px;\n\n padding: 8px;\n\n box-sizing: border-box;\n\n border: 2px solid white;\n\n display: inline-block;\n\n text-align: center;\n\n opacity: 1;\n\n visibility: visible;\n\n transition: border 0.3s ease; /* Smooth transition effect */\n\n border-left: none;\n\n border-bottom: none; /* Apply bottom border to only last item */\n\n }\n\n .social-share-list-linkedin {\n\n background-color: #0474b4;\n\n border-top-right-radius: 5px; /* Rounded top right corner of first item */\n\n }\n\n .social-share-list-facebook {\n\n background-color: #3c5c9c;\n\n }\n\n .social-share-list-xicon {\n\n background-color: #000;\n\n }\n\n .social-share-list-reddit {\n\n background-color: #fc4404;\n\n }\n\n .social-share-list-bluesky {\n\n background-color: #f0f2f5;\n\n }\n\n .social-share-list-rss {\n\n background-color: #ec7b1c;\n\n }\n\n .social-share-list-mail {\n\n background-color: #848484;\n\n border-bottom-right-radius: 5px; /* Rounded bottom right corner of last item */\n\n }\n\n .sharing-options li.social-share-list-mail {\n\n border-bottom: 2px solid white; /* Add bottom border only to the last item */\n\n height: 52px; /* Increase last child height to make in align with the hover label */\n\n }\n\n .x-icon {\n\n filter: invert(100%);\n\n transition: filter 0.3s ease;\n\n width: 20px !important;\n\n height: auto;\n\n padding-top: 5px !important;\n\n }\n\n .bluesky-icon {\n\n filter: invert(20%) sepia(100%) saturate(3000%) hue-rotate(180deg);\n\n transition: filter 0.3s ease;\n\n padding-top: 5px !important;\n\n width: 25px !important;\n\n }\n\n .share-icon {\n\n border: 2px solid transparent;\n\n display: inline-block;\n\n position: relative;\n\n }\n\n .sharing-options li:hover {\n\n border: 2px solid white;\n\n border-left: none;\n\n border-bottom: none;\n\n border-radius: 0px;\n\n }\n\n .sharing-options li.social-share-list-mail:hover {\n\n border-bottom: 2px solid white; /* Add bottom border only to the last item */\n\n }\n\n .sharing-options li:hover .label {\n\n opacity: 1;\n\n visibility: visible;\n\n border: 2px solid white;\n\n box-sizing: border-box;\n\n border-left: none;\n\n }\n\n .label {\n\n position: absolute;\n\n left: 100%;\n\n white-space: nowrap;\n\n opacity: 0;\n\n visibility: hidden;\n\n transition: all 0.2s ease;\n\n color: white;\n\n border-radius: 0 10 0 10px;\n\n top: 50%;\n\n transform: translateY(-50%);\n\n height: 52px;\n\n display: flex;\n\n align-items: center;\n\n justify-content: center;\n\n padding: 10px
```



```

LinkedIn","social.facebook.alt":"Share to Facebook","social.facebook.label":"Share on Facebook","social.x.alt":"Share to
X","social.x.label":"Share on X","social.redditt.alt":"Share to Reddit","social.redditt.label":"Share on
Reddit","social.bluesky.alt":"Share to Blue Sky","social.bluesky.label":"Share on Bluesky","social.rss.alt":"Subscribe to
RSS","social.rss.label":"Share on RSS","social.email.alt":"Share to Email","social.email.label":"Share on
Email"},"defaults":{"config":{"applicablePages":[],"description":"The Microsoft
Footer"},"fetchedContent":null,"__typename":"ComponentConfiguration"},"props":
[], "__typename":"ComponentProperties"},"components":
[{"id":"custom.widget.MicrosoftFooter","form":null,"config":null,"props":
[], "__typename":"Component"},"grouping":"CUSTOM","__typename":"ComponentTemplate"},"properties":{"config":
{"applicablePages":[],"description":"The Microsoft
Footer"},"fetchedContent":null,"__typename":"ComponentConfiguration"},"props":
[], "__typename":"ComponentProperties"},"form":null,"__typename":"Component","localOverride":false},"globalCss":
{"css": ".custom_widget_MicrosoftFooter_context-uhf_qp4x5_1 {\r\n min-width: 17.5rem;\r\n font-size: 0.9375rem;\r\n
box-sizing: border-box;\r\n -ms-text-size-adjust: 100%;\r\n -webkit-text-size-adjust: 100%;\r\n & *,\r\n & *:before,\r\n &
*:after {\r\n box-sizing: inherit;\r\n }\r\n a.custom_widget_MicrosoftFooter_c-uhff-link_qp4x5_23 {\r\n color: #616161;\r\n
word-break: break-word;\r\n text-decoration: none;\r\n }\r\n &a:link,\r\n &a:focus,\r\n &a:hover,\r\n &a:active,\r\n
&a:visited {\r\n text-decoration: none;\r\n color: inherit;\r\n }\r\n & div {\r\n font-family: 'Segoe UI', SegoeUI, Helvetica
Neue', Helvetica, Arial, sans-serif;\r\n }\r\n }\r\n .custom_widget_MicrosoftFooter_c-uhff_qp4x5_23 {\r\n background:
#f2f2f2;\r\n margin: -1.5625\r\n width: auto;\r\n height: auto;\r\n }\r\n .custom_widget_MicrosoftFooter_c-uhff-
nav_qp4x5_69 {\r\n margin: 0 auto;\r\n max-width: calc(100rem + 10%);\r\n padding: 0 5%;\r\n box-sizing: inherit;\r\n
&:before,\r\n &:after {\r\n content: '';\r\n display: table;\r\n clear: left;\r\n }\r\n @media only screen and (max-width:
1083px) {\r\n padding-left: 0.75rem;\r\n }\r\n .custom_widget_MicrosoftFooter_c-heading-4_qp4x5_97 {\r\n color:
#616161;\r\n word-break: break-word;\r\n font-size: 0.9375rem;\r\n line-height: 1.25rem;\r\n padding: 2.25rem 0
0.25rem;\r\n font-weight: 600;\r\n }\r\n .custom_widget_MicrosoftFooter_c-uhff-nav-row_qp4x5_113 {\r\n
.custom_widget_MicrosoftFooter_c-uhff-nav-group_qp4x5_115 {\r\n display: block;\r\n float: left;\r\n min-height:
0.0625rem;\r\n vertical-align: text-top;\r\n padding: 0 0.75rem;\r\n width: 100%;\r\n zoom: 1;\r\n &:first-child {\r\n padding-
left: 0;\r\n @media only screen and (max-width: 1083px) {\r\n padding-left: 0.75rem;\r\n }\r\n }\r\n @media only screen
and (min-width: 540px) and (max-width: 1082px) {\r\n width: 33.33333%;\r\n }\r\n @media only screen and (min-width:
1083px) {\r\n width: 16.6666666667%;\r\n }\r\n ul.custom_widget_MicrosoftFooter_c-
list_qp4x5_155.custom_widget_MicrosoftFooter_f-bare_qp4x5_155 {\r\n font-size: 0.6875rem;\r\n line-height: 1rem;\r\n
margin-top: 0;\r\n margin-bottom: 0;\r\n padding-left: 0;\r\n list-style-type: none;\r\n li {\r\n word-break: break-word;\r\n
padding: 0.5rem 0;\r\n margin: 0;\r\n }\r\n }\r\n }\r\n }\r\n .custom_widget_MicrosoftFooter_c-uhff-base_qp4x5_187
{\r\n background: #f2f2f2;\r\n margin: 0 auto;\r\n max-width: calc(100rem + 10%);\r\n padding: 1.875rem 5% 1rem;\r\n
&:before,\r\n &:after {\r\n content: '';\r\n display: table;\r\n }\r\n &:after {\r\n clear: both;\r\n }\r\n
a.custom_widget_MicrosoftFooter_c-uhff-ccpa_qp4x5_213,\r\n a.custom_widget_MicrosoftFooter_c-uhff-
consumer_qp4x5_215 {\r\n display: flex;\r\n float: left;\r\n font-size: 0.6875rem;\r\n line-height: 1rem;\r\n padding: 0.25rem
1.5rem 0 0;\r\n }\r\n a.custom_widget_MicrosoftFooter_c-uhff-ccpa_qp4x5_213:hover,\r\n
a.custom_widget_MicrosoftFooter_c-uhff-consumer_qp4x5_215:hover {\r\n text-decoration: underline;\r\n }\r\n
ul.custom_widget_MicrosoftFooter_c-list_qp4x5_155 {\r\n font-size: 0.6875rem;\r\n line-height: 1rem;\r\n float: right;\r\n
margin: 0.1875rem 0;\r\n color: #616161;\r\n li {\r\n padding: 0 1.5rem 0.25rem 0;\r\n display: inline-block;\r\n }\r\n }\r\n
.custom_widget_MicrosoftFooter_c-list_qp4x5_155.custom_widget_MicrosoftFooter_f-bare_qp4x5_155 {\r\n padding-left:
0;\r\n list-style-type: none;\r\n }\r\n @media only screen and (max-width: 1083px) {\r\n display: flex;\r\n flex-wrap:
wrap;\r\n padding: 1.875rem 1.5rem 1rem;\r\n }\r\n }\r\n .custom_widget_MicrosoftFooter_social-share_qp4x5_281 {\r\n
position: fixed;\r\n top: 60%;\r\n transform: translateY(-50%);\r\n left: 0;\r\n z-index:
1000;\r\n }\r\n .custom_widget_MicrosoftFooter_sharing-options_qp4x5_297 {\r\n list-style: none;\r\n padding: 0;\r\n
margin: 0;\r\n display: block;\r\n flex-direction: column;\r\n background-color: white;\r\n width: 3.125rem;\r\n border-
radius: 0 0.4375rem 0.4375rem 0;\r\n }\r\n .custom_widget_MicrosoftFooter_linkedin-icon_qp4x5_317 {\r\n border-top-
right-radius: 7px;\r\n }\r\n .custom_widget_MicrosoftFooter_linkedin-icon_qp4x5_317:hover {\r\n border-radius:
0;\r\n }\r\n .custom_widget_MicrosoftFooter_social-share-email-image_qp4x5_331:hover {\r\n border-radius:
0;\r\n }\r\n .custom_widget_MicrosoftFooter_social-link-footer_qp4x5_339:hover
.custom_widget_MicrosoftFooter_linkedin-icon_qp4x5_317 {\r\n border-radius:
0;\r\n }\r\n .custom_widget_MicrosoftFooter_social-link-footer_qp4x5_339:hover .custom_widget_MicrosoftFooter_social-
share-email-image_qp4x5_331 {\r\n border-radius: 0;\r\n }\r\n .custom_widget_MicrosoftFooter_social-link-
footer_qp4x5_339 img {\r\n width: 1.875rem;\r\n height: auto;\r\n transition: filter 0.3s
ease;\r\n }\r\n .custom_widget_MicrosoftFooter_social-share-list_qp4x5_365 {\r\n width:
3.125rem;\r\n }\r\n .custom_widget_MicrosoftFooter_social-share-rss-image_qp4x5_371 {\r\n width: 1.875rem;\r\n height:
auto;\r\n transition: filter 0.3s ease;\r\n }\r\n .custom_widget_MicrosoftFooter_sharing-options_qp4x5_297 li {\r\n width:
3.125rem;\r\n height: 3.125rem;\r\n padding: 0.5rem;\r\n box-sizing: border-box;\r\n border: 2px solid white;\r\n display:
inline-block;\r\n text-align: center;\r\n opacity: 1;\r\n visibility: visible;\r\n transition: border 0.3s ease; /* Smooth transition
effect */\r\n border-left: none;\r\n border-bottom: none; /* Apply bottom border to only last item
*/\r\n }\r\n .custom_widget_MicrosoftFooter_social-share-list-linkedin_qp4x5_411 {\r\n background-color: #0474b4;\r\n
border-top-right-radius: 5px; /* Rounded top right corner of first item*/\r\n }\r\n .custom_widget_MicrosoftFooter_social-
share-list-facebook_qp4x5_419 {\r\n background-color: #3c5c9c;\r\n }\r\n .custom_widget_MicrosoftFooter_social-share-
list-xicon_qp4x5_425 {\r\n background-color: #000;\r\n }\r\n .custom_widget_MicrosoftFooter_social-share-list-

```


spam","messageMarkedAsSpam@board:OCCASION":"This event has been marked as spam",

spam","messageMarkedAsSpam@board:IDEA":"This idea has been marked as spam","manageSpam":"Manage Spam",

messageMarkedAsAbuse":"This post has been marked as abuse",messageMarkedAsAbuse@board:TKB":"This article has been marked as abuse",

messageMarkedAsAbuse@board:BLOG":"This post has been marked as abuse",

messageMarkedAsAbuse@board:FORUM":"This discussion has been marked as abuse",

messageMarkedAsAbuse@board:OCCASION":"This event has been marked as abuse",

messageMarkedAsAbuse@board:IDEA":"This idea has been marked as abuse",

preModCommentAuthorText":"This comment will be published as soon as it is approved",

preModCommentModeratorText":"This comment is awaiting moderation",

messageMarkedAsOther":"This post has been rejected due to other reasons",

messageMarkedAsOther@board:TKB":"This article has been rejected due to other reasons",

messageMarkedAsOther@board:BLOG":"This post has been rejected due to other reasons",

messageMarkedAsOther@board:FORUM":"This discussion has been rejected due to other reasons",

messageMarkedAsOther@board:OCCASION":"This event has been rejected due to other reasons",

messageMarkedAsOther@board:IDEA":"This idea has been rejected due to other reasons",

messageArchived":"This post was archived on {date}","relatedUrl":"View Related Content",

relatedContentText":"Showing related content",

archivedContentLink":"View Archived Content"},

localOverride:false,"Category:category:Exchange":

{ "__typename":"Category","id":"category:Exchange","categoryPolicies":

{ "__typename":"CategoryPolicies","canReadNode":

{ "__typename":"PolicyResult","failureReason":null } },"Category:category:Outlook":

{ "__typename":"Category","id":"category:Outlook","categoryPolicies":{ "__typename":"CategoryPolicies","canReadNode":

{ "__typename":"PolicyResult","failureReason":null } },"Category:category:Community-Info-Center":

{ "__typename":"Category","id":"category:Community-Info-Center","categoryPolicies":

{ "__typename":"CategoryPolicies","canReadNode":

{ "__typename":"PolicyResult","failureReason":null } },"Category:category:EducationSector":

{ "__typename":"Category","id":"category:EducationSector","categoryPolicies":

{ "__typename":"CategoryPolicies","canReadNode":

{ "__typename":"PolicyResult","failureReason":null } },"Category:category:DrivingAdoption":

{ "__typename":"Category","id":"category:DrivingAdoption","categoryPolicies":

{ "__typename":"CategoryPolicies","canReadNode":

{ "__typename":"PolicyResult","failureReason":null } },"Category:category:Azure":

{ "__typename":"Category","id":"category:Azure","categoryPolicies":{ "__typename":"CategoryPolicies","canReadNode":

{ "__typename":"PolicyResult","failureReason":null } },"Category:category:Windows-Server":

{ "__typename":"Category","id":"category:Windows-Server","categoryPolicies":

{ "__typename":"CategoryPolicies","canReadNode":

{ "__typename":"PolicyResult","failureReason":null } },"Category:category:MicrosoftTeams":

{ "__typename":"Category","id":"category:MicrosoftTeams","categoryPolicies":

{ "__typename":"CategoryPolicies","canReadNode":

{ "__typename":"PolicyResult","failureReason":null } },"Category:category:PublicSector":

{ "__typename":"Category","id":"category:PublicSector","categoryPolicies":

{ "__typename":"CategoryPolicies","canReadNode":

{ "__typename":"PolicyResult","failureReason":null } },"Category:category:microsoft365":

{ "__typename":"Category","id":"category:microsoft365","categoryPolicies":

{ "__typename":"CategoryPolicies","canReadNode":

{ "__typename":"PolicyResult","failureReason":null } },"Category:category:IoT":

{ "__typename":"Category","id":"category:IoT","categoryPolicies":{ "__typename":"CategoryPolicies","canReadNode":

{ "__typename":"PolicyResult","failureReason":null } },"Category:category:HealthcareAndLifeSciences":

{ "__typename":"Category","id":"category:HealthcareAndLifeSciences","categoryPolicies":

{ "__typename":"CategoryPolicies","canReadNode":

{ "__typename":"PolicyResult","failureReason":null } },"Category:category:ITOpsTalk":

{ "__typename":"Category","id":"category:ITOpsTalk","categoryPolicies":

{ "__typename":"CategoryPolicies","canReadNode":

{ "__typename":"PolicyResult","failureReason":null } },"Category:category:MicrosoftMechanics":

{ "__typename":"Category","id":"category:MicrosoftMechanics","categoryPolicies":

{ "__typename":"CategoryPolicies","canReadNode":

{ "__typename":"PolicyResult","failureReason":null } },"Category:category:MicrosoftforNonprofits":

{ "__typename":"Category","id":"category:MicrosoftforNonprofits","categoryPolicies":

{ "__typename":"CategoryPolicies","canReadNode":

{ "__typename":"PolicyResult","failureReason":null } },"Category:category:PartnerCommunity":

{ "__typename":"Category","id":"category:PartnerCommunity","categoryPolicies":

{ "__typename":"CategoryPolicies","canReadNode":

{ "__typename":"PolicyResult","failureReason":null } },"Category:category:Microsoft365Copilot":

{ "__typename":"Category","id":"category:Microsoft365Copilot","categoryPolicies":

{ "__typename":"CategoryPolicies","canReadNode":

```
{ "__typename": "PolicyResult", "failureReason": null } }, "Category:category:Windows":  
{ "__typename": "Category", "id": "category:Windows", "categoryPolicies":  
{ "__typename": "CategoryPolicies", "canReadNode":  
{ "__typename": "PolicyResult", "failureReason": null } }, "Category:category:Content_Management":  
{ "__typename": "Category", "id": "category:Content_Management", "categoryPolicies":  
{ "__typename": "CategoryPolicies", "canReadNode":  
{ "__typename": "PolicyResult", "failureReason": null } }, "Category:category:CommunityNewsDesk":  
{ "__typename": "Category", "id": "category:CommunityNewsDesk", "categoryPolicies":  
{ "__typename": "CategoryPolicies", "canReadNode":  
{ "__typename": "PolicyResult", "failureReason": null } }, "Category:category:microsoft-learn-for-educators":  
{ "__typename": "Category", "id": "category:microsoft-learn-for-educators", "categoryPolicies":  
{ "__typename": "CategoryPolicies", "canReadNode":  
{ "__typename": "PolicyResult", "failureReason": null } }, "Category:category:mvp":  
{ "__typename": "Category", "id": "category:mvp", "categoryPolicies": { "__typename": "CategoryPolicies", "canReadNode":  
{ "__typename": "PolicyResult", "failureReason": null } }, "Category:category:microsoftintune":  
{ "__typename": "Category", "id": "category:microsoftintune", "categoryPolicies":  
{ "__typename": "CategoryPolicies", "canReadNode":  
{ "__typename": "PolicyResult", "failureReason": null } }, "Category:category:microsoft-global-community-initiative":  
{ "__typename": "Category", "id": "category:microsoft-global-community-initiative", "categoryPolicies":  
{ "__typename": "CategoryPolicies", "canReadNode":  
{ "__typename": "PolicyResult", "failureReason": null } }, "Category:category:usergroups":  
{ "__typename": "Category", "id": "category:usergroups", "categoryPolicies":  
{ "__typename": "CategoryPolicies", "canReadNode":  
{ "__typename": "PolicyResult", "failureReason": null } }, "Category:category:skills-hub":  
{ "__typename": "Category", "id": "category:skills-hub", "categoryPolicies":  
{ "__typename": "CategoryPolicies", "canReadNode":  
{ "__typename": "PolicyResult", "failureReason": null } }, "Blog:board:skills-hub-blog":  
{ "__typename": "Blog", "id": "board:skills-hub-blog", "blogPolicies": { "__typename": "BlogPolicies", "canReadNode":  
{ "__typename": "PolicyResult", "failureReason": null } }, "boardPolicies": { "__typename": "BoardPolicies", "canReadNode":  
{ "__typename": "PolicyResult", "failureReason": null } }, "CachedAsset:text:en_US-components/community/Navbar-  
1775111750899": { "__typename": "CachedAsset", "id": "text:en_US-components/community/Navbar-  
1775111750899", "value": { "community": "Community Home", "inbox": "Inbox", "manageContent": "Manage  
Content", "tos": "Terms of Service", "forgotPassword": "Forgot Password", "themeEditor": "Theme Editor", "edit": "Edit  
Navigation Bar", "skipContent": "Skip to content", "gxcuf89792": "Tech Community", "windows-server": "Windows  
Server", "ms-learn-ext-security": "Microsoft Security", "Common_Enntvz-i-t-ops-talk-link": "ITOps Talk", "education-  
sector": "Education Sector", "Common-external-link-9": "Microsoft 365", "Common-external-link-8": "Dynamics  
365", "Common-external-link-7": "Skilling Room Directory", "Common-external-link-6": "Events", "Common-external-link-  
5": "Blogs", "Common-external-link-4": "View All", "Common-gxcuf89792-community": "Community", "Common-external-  
link-3": "Topics", "microsoft365": "Microsoft 365", "Common_Enntvz-community-news-desk-link": "Community News  
Desk", "Common_Enntvz-azure-link": "Azure", "Common-community-info-center-  
link": "Lounge", "azure": "Azure", "Common_Enntvz-windows-link": "Windows", "Common_Enntvz-education-sector-  
link": "Education Sector", "Common-windows-server-link": "Windows Server", "products-  
link": "Products", "Common_Enntvz-partner-community-link": "Microsoft Partner Community", "microsoft-learn-  
blog": "Blog", "Common-external-link-2": "View All", "community-hub-link": "Community Hubs", "Common-mvp-  
link": "Microsoft MVP Program", "community-info-center": "Lounge", "microsoft-endpoint-manager": "Microsoft  
Intune", "startupsat-microsoft": "Startups at Microsoft", "ms-learn-ext-azure": "Azure", "Common_Enntvz-  
content_management-link": "Content Management", "ms-learn-ext-github": "Github", "Common-microsoft365-  
link": "Microsoft 365", "Common-i-t-ops-talk-link": "ITOps Talk", "Common_Enntvz-view-all-products-link": "View  
All", "Common-microsoft-global-community-initiative-link": "Microsoft Global Community Initiative (MGCI)", "all-events-  
link": "Events", "Common_Enntvz-microsoft-learn-for-educators-link": "Microsoft Learn for Educators", "Common-external-  
link": "Community Hubs", "Common-partner-community-link": "Microsoft Partner Community", "Common-microsoft-learn-  
for-educators-link": "Microsoft Learn for Educators", "Common_Enntvz-microsoft-teams-link": "Microsoft Teams", "driving-  
adoption": "Driving Adoption", "microsoft-learn": "Microsoft Learn", "Common-healthcare-and-life-sciences-  
link": "Healthcare and Life Sciences", "planner": "Outlook", "Common_Enntvz-exchange-link": "Exchange", "healthcare-and-  
life-sciences": "Healthcare and Life Sciences", "Common-external-link-10": "View All", "Common-driving-adoption-  
link": "Driving Adoption", "ms-learn-ext-pp": "Power Platform", "Common_Enntvz-windows-server-link": "Windows  
Server", "Common-io-t-link": "Internet of Things (IoT)", "Skills-Hub": "Skills Hub", "microsoft-teams": "Microsoft  
Teams", "Common-outlook-link": "Outlook", "Common_Enntvz-public-sector-link": "Public Sector", "Common-windows-  
link": "Windows", "all-blogs-link": "Blogs", "communities": "Products", "Common_Enntvz-usergroups-link": "User  
Groups", "Common_Enntvz-microsoft-global-community-initiative-link": "Microsoft Global Community Initiative  
(MGCI)", "Skills-Hub-link": "Community", "Common_Enntvz-io-t-link": "Internet of Things (IoT)", "ms-learn-ext-  
m365": "Microsoft 365", "Common_Enntvz-microsoft-mechanics-link": "Microsoft Mechanics", "microsoft-learn-  
community": "Community", "partner-community": "Microsoft Partner Community", "Common-microsoft-mechanics-  
link": "Microsoft Mechanics", "Common_Enntvz-healthcare-and-life-sciences-link": "Healthcare and Life
```

Sciences", "microsoft-mechanics": "Microsoft Mechanics", "Common-microsoft-security-link": "Microsoft Security", "Common-education-sector-link": "Education Sector", "Skills-Hub-Blog": "Blog", "i-t-ops-talk": "ITOps Talk", "microsoft-securityand-compliance": "Microsoft Security", "Common_Enntvz-microsoftintune-link": "Microsoft Intune", "Common-azure-link": "Azure", "Common-microsoftintune-link": "Microsoft Intune", "Common_Enntvz-view-all-topics-link": "View All", "Common-usergroups-link": "User Groups", "Common-public-sector-link": "Public Sector", "Common_Enntvz-microsoft-security-link": "Microsoft Security", "Common_Enntvz-outlook-link": "Outlook", "Common_Enntvz-mvp-link": "Microsoft MVP Program", "exchange": "Exchange", "topics-link": "Topics", "io-t": "Internet of Things (IoT)", "Common-microsoft365-copilot-link": "Microsoft 365 Copilot", "Common-microsoft-teams-link": "Microsoft Teams", "s-m-b": "Nonprofit Community", "Common_Enntvz-community-info-center-link": "Lounge", "Common_Enntvz-microsoft365-copilot-link": "Microsoft 365 Copilot", "Common_Enntvz-microsoftfor-nonprofits-link": "Nonprofit Community", "Common_Enntvz-microsoft365-link": "Microsoft 365", "Common-content_management-link": "Content Management", "ms-learn-ext-teams": "Teams", "s-q-l-server": "Content Management", "products-services": "Products", "Common-community-news-desk-link": "Community News Desk", "ms-learn-ext-LD": "Skilling Room Directory", "Common-exchange-link": "Exchange", "Common-gxcuf89792-link": "Tech Community", "windows": "Windows", "public-sector": "Public Sector", "Common_Enntvz-driving-adoption-link": "Driving Adoption", "Common-microsoftfor-nonprofits-link": "Nonprofit Community", "ms-learn-ext-net": ".NET", "ms-learn-ext-dynamics": "Dynamics 365", "a-i": "AI and Machine Learning", "outlook": "Microsoft 365 Copilot", "localOverride": false, "CachedAsset:text:en_US-components/community/NavbarHamburgerDropdown-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/community/NavbarHamburgerDropdown-1775111750899", "value": {"hamburgerLabelOpen": "Open Side Menu", "hamburgerLabelClose": "Close Side Menu"}, "localOverride": false, "CachedAsset:text:en_US-components/community/BrandLogo-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/community/BrandLogo-1775111750899", "value": {"logoAlt": "Khoros", "themeLogoAlt": "Brand Logo", "linkAriaLabel": "Go to community home page"}, "localOverride": false, "CachedAsset:text:en_US-components/community/NavbarTextLinks-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/community/NavbarTextLinks-1775111750899", "value": {"more": "More"}, "localOverride": false, "CachedAsset:text:en_US-components/search/SpotlightSearchIcon-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/search/SpotlightSearchIcon-1775111750899", "value": {"search": "Search"}, "localOverride": false, "CachedAsset:text:en_US-components/authentication/AuthenticationLink-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/authentication/AuthenticationLink-1775111750899", "value": {"title.login": "Sign In", "title.registration": "Register", "title.forgotPassword": "Forgot Password", "title.multiAuthLogin": "Sign In"}, "localOverride": false, "CachedAsset:text:en_US-components/nodes/NodeLink-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/nodes/NodeLink-1775111750899", "value": {"place": "Go back to {name}"}, "localOverride": false, "CachedAsset:text:en_US-components/messages/MessageView/MessageViewStandard-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/messages/MessageView/MessageViewStandard-1775111750899", "value": {"anonymous": "Anonymous", "author": "messageAuthorLogin", "authorBy": "messageAuthorLogin", "board": "messageBoardTitle", "replyToUser": "to {parentAuthor}", "showMoreReplies": "Show More", "replyText": "Reply", "repliesText": "Replies", "markedAsSolved": "Marked as Solution", "messageStatus": "Status", "statusChanged": "Status changed: {previousStatus} to {currentStatus}", "statusAdded": "Status added: {status}", "statusRemoved": "Status removed: {status}", "labelExpand": "expand replies", "labelCollapse": "collapse replies", "unhelpfulReason.reason1": "Content is outdated", "unhelpfulReason.reason2": "Article is missing information", "unhelpfulReason.reason3": "Content is for a different Product", "unhelpfulReason.reason4": "Doesn't match what I was searching for"}, "localOverride": false, "CachedAsset:text:en_US-components/messages/MessageReplyCallToAction-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/messages/MessageReplyCallToAction-1775111750899", "value": {"leaveReply": "Leave a reply...", "leaveReply@board: BLOG@message:root": "Leave a comment...", "leaveReply@board:TKB@message:root": "Leave a comment...", "leaveReply@board:IDEA@message:root": "Leave a comment...", "leaveReply@board:OCCASION@message:root": "Leave a comment...", "repliesTurnedOff.FORUM": "Replies are turned off for this topic", "repliesTurnedOff.BLOG": "Comments are turned off for this topic", "repliesTurnedOff.TKB": "Comments are turned off for this topic", "repliesTurnedOff.IDEA": "Comments are turned off for this topic", "repliesTurnedOff.OCCASION": "Comments are turned off for this topic", "infoText": "Stop poking me!"}, "localOverride": false, "CachedAsset:text:en_US-components/community/NavbarDropdownToggle-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/community/NavbarDropdownToggle-1775111750899", "value": {"ariaLabelClosed": "Press the down arrow to open the menu"}, "localOverride": false, "CachedAsset:text:en_US-components/messages/MessageCoverImage-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/messages/MessageCoverImage-1775111750899", "value": {"coverImageTitle": "Cover Image"}, "localOverride": false, "CachedAsset:text:en_US-shared/client/components/nodes/NodeTitle-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-shared/client/components/nodes/NodeTitle-1775111750899", "value": {"nodeTitle": "nodeTitle, select, community {Community} other {nodeTitle}"}, "localOverride": false, "CachedAsset:text:en_US-components/messages/MessageTimeToRead-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/messages/MessageTimeToRead-1775111750899", "value": {"minReadText": "{min} MIN

```
READ"), "localOverride": false}, "CachedAsset": {text: "en_US-components/messages/MessageSubject-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/messages/MessageSubject-1775111750899", "value": {"noSubject": "no subject"}, "localOverride": false}, "CachedAsset": {text: "en_US-components/users/UserLink-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/users/UserLink-1775111750899", "value": {"authorName": "View Profile: {author}", "anonymous": "Anonymous", "ariaLabel.rank": "Rank: {rankName}"}, "localOverride": false}, "CachedAsset": {text: "en_US-shared/client/components/users/UserRank-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-shared/client/components/users/UserRank-1775111750899", "value": {"rankName": "{rankName}", "userRank": "Author rank {rankName}"}, "localOverride": false}, "CachedAsset": {text: "en_US-components/messages/MessageTime-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/messages/MessageTime-1775111750899", "value": {"postTime": "Published: {time}", "lastPublishTime": "Last Update: {time}", "conversation.lastPostingActivityTime": "Last posting activity time: {time}", "conversation.lastPostTime": "Last post time: {time}", "moderationData.rejectTime": "Rejected time: {time}"}, "localOverride": false}, "CachedAsset": {text: "en_US-components/messages/MessageBody-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/messages/MessageBody-1775111750899", "value": {"showMessageBody": "Show More", "mentionsErrorTitle": "{mentionsType, select, board {Board} user {User} message {Message} other {} No Longer Available", "mentionsErrorMessage": "The {mentionsType} you are trying to view has been removed from the community.", "videoProcessing": "Video is being processed. Please try again in a few minutes.", "bannerTitle": "Video provider requires cookies to play the video. Accept to continue or {url} it directly on the provider's site.", "buttonTitle": "Accept", "urlText": "watch"}, "localOverride": false}, "CachedAsset": {text: "en_US-components/messages/MessageCustomFields-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/messages/MessageCustomFields-1775111750899", "value": {"CustomField.default.label": "Value of {name}"}, "localOverride": false}, "CachedAsset": {text: "en_US-components/messages/MessageRevision-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/messages/MessageRevision-1775111750899", "value": {"lastUpdatedDatePublished": "{publishCount, plural, one {Published} other {Updated}} {date}", "lastUpdatedDateDraft": "Created {date}", "version": "Version {major} {minor}"}, "localOverride": false}, "CachedAsset": {text: "en_US-shared/client/components/common/QueryHandler-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-shared/client/components/common/QueryHandler-1775111750899", "value": {"title": "Query Handler"}, "localOverride": false}, "CachedAsset": {text: "en_US-components/tags/TagList-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/tags/TagList-1775111750899", "value": {"showMoreFor": "Show more for {title}"}, "localOverride": false}, "CachedAsset": {text: "en_US-components/messages/MessageReplyButton-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/messages/MessageReplyButton-1775111750899", "value": {"repliesCount": "{count}", "title": "Reply", "title@board: BLOG@message:root": "Comment", "title@board: TKB@message:root": "Comment", "title@board: IDEA@message:components/messages/MessageAuthorBio-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/messages/MessageAuthorBio-1775111750899", "value": {"sendMessage": "Send Message", "actionMessage": "Follow this blog board to get notified when there's new activity", "coAuthor": "CO-PUBLISHER", "contributor": "CONTRIBUTOR", "userProfile": "View Profile", "iconlink": "Go to {name} {type}"}, "localOverride": false}, "CachedAsset": {text: "en_US-shared/client/components/users/UserAvatar-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-shared/client/components/users/UserAvatar-1775111750899", "value": {"altText": "{login}'s avatar", "altTextGeneric": "User's avatar"}, "localOverride": false}, "CachedAsset": {text: "en_US-shared/client/components/ranks/UserRankLabel-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-shared/client/components/ranks/UserRankLabel-1775111750899", "value": {"altTitle": "Icon for {rankName} rank"}, "localOverride": false}, "CachedAsset": {text: "en_US-components/tags/TagView/TagViewChip-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/tags/TagView/TagViewChip-1775111750899", "value": {"tagLabelName": "Tag name {tagName}"}, "localOverride": false}, "CachedAsset": {text: "en_US-components/users/UserRegistrationDate-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-components/users/UserRegistrationDate-1775111750899", "value": {"noPrefix": "{date}", "withPrefix": "Joined {date}"}, "localOverride": false}, "CachedAsset": {text: "en_US-shared/client/components/nodes/NodeAvatar-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-shared/client/components/nodes/NodeAvatar-1775111750899", "value": {"altTitle": "Node avatar for {nodeTitle}"}, "localOverride": false}, "CachedAsset": {text: "en_US-shared/client/components/nodes/NodeDescription-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-shared/client/components/nodes/NodeDescription-1775111750899", "value": {"description": "{description}"}, "localOverride": false}, "CachedAsset": {text: "en_US-shared/client/components/nodes/NodeIcon-1775111750899": {"__typename": "CachedAsset", "id": "text:en_US-shared/client/components/nodes/NodeIcon-1775111750899", "value": {"contentType": "Content Type {style, select, FORUM {Forum} BLOG {Blog} TKB {Knowledge Base} IDEA {Ideas} OCCASION {Events} other {} icon"}, "localOverride": false}}}, "page": "/blogs/BlogMessagePage/BlogMessagePage", "query": {"boardId": "microsoftsentinelblog", "messageSubject": "solarwinds-post-compromise-hunting-with-azure-sentinel", "messageId": "1995095"}, "buildId": "VXuOn2D5MfObWEiRanLQ9", "runtimeConfig": {"buildInformationVisible": false, "logLevelApp": "info", "logLevelMetrics": "info", "surveysEnabled": true, "openTelemetry": {"clientEnabled": false, "configName": "o365", "serviceVersion": "26.1.0", "universe": "prod", "collector": "http://localhost:4318", "logLevel": "error", "routeCh": ["components_community_Navbar_NavbarWidget", "components_community_Breadcrumb_BreadcrumbWidget", "components_customComponent_Cust ["id": "analytics", "src": "https://techcommunity.microsoft.com/t5/s/gxcuf89792/pagescripts/1751476272000/analytics.js? page.id=BlogMessagePage&entity.id=board%3Amicrosoftsentinelblog&entity.id=message%3A1995095", "strategy": "afterInteractive"]}}
```

Source: <https://techcommunity.microsoft.com/t5/azure-sentinel/solarwinds-post-compromise-hunting-with-azure-sentinel/ba-p/1995095>