



Appendix

Original Prompt and Inputted App Architecture

Create an attack tree diagram for the following app architecture, with the top node being stealing user data. App architecture: User launches the app and logs in or registers via Amazon Cognito. Game assets required for play are fetched from Amazon S3. As the user plays, their game state (score, resources, etc.) is continuously updated in DynamoDB. Certain in-game events trigger AWS Lambda functions for processing. If users participate in multiplayer events, GameLift ensures seamless gameplay. Offline plays are synced back to DynamoDB using AppSync once the user is online. User behavior and game interactions are continuously sent to AWS Analytics for evaluation and insights. Amazon Pinpoint engages users with timely and relevant push notifications.

Usage Costs

Total Cost: \$0.023969999999999998 USD

Input Tokens Cost: \$0.0 USD

Output Tokens Cost: \$0.023969999999999998 USD

Conversation Log

| STARTING CONVERSATION: Create an attack tree diagram for the following app architecture, with the top node being stealing user data. App architecture: User launches the app and logs in or registers via Amazon Cognito. Game assets required for play are fetched from Amazon S3. As the user plays, their game state (score, resources, etc.) is continuously updated in DynamoDB. Certain in-game events trigger AWS Lambda functions for processing. If users participate in multiplayer events, GameLift ensures seamless gameplay. Offline plays are synced back to DynamoDB using AppSync once the user is online. User behavior and game interactions are continuously sent to AWS Analytics for evaluation and insights. Amazon Pinpoint engages users with timely and relevant push notifications.DESCRPTION_END | user_proxy (to chatbot): | | Create an attack tree diagram for the following app architecture, with the top node being stealing user data. App architecture: User launches the app and logs in or registers via Amazon Cognito. Game assets required for play are fetched from Amazon S3. As the user plays, their game state (score, resources, etc.) is continuously updated in DynamoDB. Certain in-game events trigger AWS Lambda functions for processing. If users participate in multiplayer events, GameLift ensures seamless gameplay. Offline plays are synced back to DynamoDB using AppSync once the user is online. User behavior and game interactions are continuously sent to AWS Analytics for evaluation and insights. Amazon Pinpoint engages users with timely and relevant push notifications.DESCRPTION_END | | ----- | chatbot (to user_proxy): | | * **Suggested function Call:** python | **Arguments:** | { | "cell": "# filename: attack_tree.py\n\nfrom graphviz import Digraph\n\n# Create a directed graph\nndot = Digraph(comment='Attack Tree')\nndot.attr('graph', margin='0', pad='0')\n\n# Add nodes for each potential attack vector\nndot.node('A', '"Stealing User's Data"', color='red')\nndot.node('B', 'Amazon Cognito', color='lightblue')\nndot.node('C', 'Amazon S3', color='lightblue')\nndot.node('D', 'DynamoDB', color='lightblue')\nndot.node('E', 'AWS Lambda', color='lightblue')\nndot.node('F', 'GameLift',

```

color='lightblue')\ndot.node('G', 'AppSync', color='lightblue')\ndot.node('H', 'AWS Analytics',
color='lightblue')\ndot.node('I', 'Amazon Pinpoint', color='lightblue')\n\n# Add edges to represent the
attack vectors\ndot.edge('A', 'B', label='Intercept login/registration data')\ndot.edge('A', 'C',
label='Intercept game assets')\ndot.edge('A', 'D', label='Manipulate game state data')\ndot.edge('A', 'E',
label='Exploit Lambda functions')\ndot.edge('A', 'F', label='Attack multiplayer events')\ndot.edge('A',
'G', label='Intercept offline plays data')\ndot.edge('A', 'H', label='Access user behavior data')
\ndot.edge('A', 'I', label='Send malicious notifications')\n\n# Save the graph as a SVG file\ndot.format =
'svg'\ndot.render('attack_tree', view=False)\nTERMINATE"      |    }    |    *****    |    |
----- | | >>>>>>> EXECUTING FUNCTION python...
|

```