AWS System Manager

AWS Systems Manager is a comprehensive management service that enables you to manage your AWS resources and on-premises systems more securely and at scale. It simplifies resource and application management by offering a unified user interface, so you can view operational data from multiple AWS services, automate tasks across your AWS resources, and maintain security and compliance.

Key Features of AWS Systems Manager:

1. **Operational Insights:**
    - **Resource Groups:** Create logical groups of resources such as EC2 instances, S3 buckets, or Lambda functions. This makes it easier to manage and monitor them collectively.
    - **Explorer:** Provides a unified view of your operational data across AWS accounts and regions, including insights into account security, operations, and resources.
    - **OpsCenter:** Centralizes the handling of operational issues by aggregating and correlating OpsItems (operational work items) from multiple AWS services.
2. **Automation:**
    - **Run Command:** Allows you to execute scripts or commands on your EC2 instances and on-premises servers without logging in, enabling you to automate tasks such as software updates and configurations.
    - **Automation:** Provides predefined workflows for common maintenance and deployment tasks, such as creating AMIs or rebooting instances. You can also create custom workflows.
3. **Patch Management:**
    - **Patch Manager:** Automates the process of patching managed instances with security updates. It supports Windows and Linux systems, ensuring they remain compliant with patching policies.
4. **Inventory and Compliance:**
    - **Inventory:** Collects detailed information about your instances and the software installed on them. This helps in tracking configurations and ensuring compliance.
    - **Compliance:** Monitors your resource configurations and compares them to desired states defined in your policies. It can automatically remediate non-compliant resources.
5. **Parameter Store:**

    - A centralized store to manage configuration data, secrets, and licensing information in a secure, scalable, and organized way. Parameters can be encrypted with AWS Key Management Service (KMS).
6. **Session Manager:**
    - Provides a secure, auditable, and browser-based interactive shell to connect to your EC2 instances and on-premises servers. It eliminates the need to open inbound ports, manage SSH keys, or use bastion hosts.
7. **State Manager:**
    - Helps you define and maintain consistent configurations on your Amazon EC2 instances and on-premises servers. It automatically enforces desired states by applying configuration scripts or other management commands.
8. **Maintenance Windows:**
    - Allows you to define specific time periods for executing administrative and maintenance tasks across your instances, helping minimize disruption to your services.

Use Cases for AWS Systems Manager:

- **Centralized Management:** Manage and automate a wide range of tasks across AWS resources and on-premises systems from a single interface.
- **Security and Compliance:** Keep your systems updated and compliant with security policies by automating patch management and configuration enforcement.
- **Operational Efficiency:** Simplify and automate common administrative tasks, such as software installations, updates, and resource provisioning.

Integration with Other AWS Services:

AWS Systems Manager integrates with other AWS services like Amazon CloudWatch, AWS Config, AWS CloudTrail, and AWS Identity and Access Management (IAM), enabling seamless management and monitoring of your AWS environment.

This tool is particularly useful for DevOps teams, system administrators, and anyone involved in managing and securing AWS resources and on-premises infrastructure.

Setup step by step

Setting up AWS Systems Manager involves several steps, from preparing your environment to configuring specific features. Here's a step-by-step guide to get you started:

Step 1: Prepare Your Environment

1. **AWS Account:**
    - Ensure you have an AWS account. If not, you can create one here.
2. **IAM Roles and Permissions:**
    - **For EC2 Instances:** You need to create an IAM role that grants the necessary permissions for Systems Manager to manage your instances.
        - Go to the **IAM Console** > **Roles** > **Create Role**.
        - Select **AWS Service** as the trusted entity and choose **EC2**.
        - Attach the following policies:
            - **AmazonEC2RoleforSSM**: Provides Systems Manager with permissions to access EC2 instances.
            - **AmazonSSMManagedInstanceCore**: Grants Systems Manager the necessary permissions to manage EC2 instances.
        - Name the role (e.g., `SSM-EC2-Role`) and create it.
        - Attach this role to your EC2 instances by selecting the instance in the EC2 console, choosing **Actions** > **Security** > **Modify IAM Role**, and assigning the role you just created.
3. **SSM Agent:**
    - The SSM Agent is pre-installed on Amazon Linux, Amazon Linux 2, Ubuntu Server 16.04 or later, and Windows Server 2016 or later AMIs. For other operating systems, you might need to manually install it.
    - To manually install the SSM Agent:
        - For Linux:
          ```
          sudo yum install -y amazon-ssm-agent
          ```
        - For Windows, download the installer from the AWS website and run it.

Step 2: Enable AWS Systems Manager

1. **Register Your Instances:**
    - Go to the **Systems Manager Console** > **Managed Instances**.
    - Ensure that your EC2 instances appear in the list. If not, check the IAM role and SSM Agent installation.
2. **Set Up Resource Groups:**
    - Go to **Resource Groups** in the Systems Manager Console.
    - Create resource groups based on tags or specific criteria for easier management.

Step 3: Configure Key Systems Manager Features

1. **Run Command:**
    - Go to **Run Command** in the Systems Manager Console.
    - Choose **Run Command** and select a predefined command document (e.g., `AWS-RunShellScript` for Linux or `AWS-RunPowerShellScript` for Windows).
    - Choose the instances on which you want to run the command, input the command or script, and execute it.
2. **Patch Manager:**
    - Go to **Patch Manager** in the Systems Manager Console.
    - Create patch baselines to define the patches that should be applied to your instances.
    - Use the **Patch Now** feature to manually trigger patching or schedule regular patching operations.
3. **Parameter Store:**
    - Go to **Parameter Store** in the Systems Manager Console.
    - Create a new parameter to store configuration data, secrets, or other sensitive information.
    - Parameters can be accessed using the Systems Manager API, CLI, or SDKs.
4. **Session Manager:**
    - Go to **Session Manager** in the Systems Manager Console.
    - Click **Start session**, select the instance you want to connect to, and initiate a session.
    - You can now interact with the instance via a browser-based shell or the AWS CLI.
5. **Automation:**
    - Go to **Automation** in the Systems Manager Console.
    - Choose **Execute Automation** and select a predefined automation document (e.g., `AWS-UpdateLinuxAmi`).
    - Provide the required inputs, select the target instances or resources, and start the automation.
6. **Compliance and Inventory:**
    - Go to **Compliance** and **Inventory** in the Systems Manager Console.
    - Set up rules to monitor resource compliance against your defined configurations and view inventory data of your managed instances.

Step 4: Monitor and Manage Your Environment

1. **Explorer:**
    - Use **Explorer** in the Systems Manager Console to get a high-level view of your environment, including security and compliance status, operational issues, and more.
2. **OpsCenter:**
    - Use **OpsCenter** to view, investigate, and resolve operational issues across your AWS environment.
3. **Maintenance Windows:**
    - Schedule maintenance tasks using **Maintenance Windows** to automate regular tasks like patching or script execution during off-peak hours.

Step 5: Advanced Configuration

1. **Integrate with Other AWS Services:**
    - You can integrate AWS Systems Manager with services like AWS CloudWatch for monitoring, AWS Config for configuration management, and AWS CloudTrail for auditing.
2. **Custom Automation Documents:**
    - Create custom automation documents using the JSON or YAML format to define your specific workflows.

By following these steps, you'll have AWS Systems Manager set up and configured to manage your AWS resources and on-premises infrastructure efficiently.