## Scenario-Based ELK Questions and Answers for Cybersecurity (Shards, Indices, ILM, Index Templates)

### 🔹 1. Shards and Indexing

**Scenario 1: Handling High Ingestion**

You are receiving logs at 100K events per second from multiple firewalls. Elasticsearch starts showing high heap usage and slow search performance.

**Q1:** How would you check if the shard distribution is optimal across nodes?

**Answer:**

* Use the \_cat/shards or \_cat/allocation APIs to review shard distribution.
* Validate even distribution across nodes, checking for any node holding too many primary or replica shards.
* Monitor node-level metrics in Kibana (e.g., node heap usage, indexing rate).

**Q2:** What shard size would you consider optimal and why?

**Answer:**

* Ideal shard size is 20–50 GB, depending on use case and hardware.
* Too small shards cause cluster state bloat; too large shards lead to performance bottlenecks during merges and recovery.

**Q3:** How would you reindex or adjust the number of shards without downtime?

**Answer:**

* Use the \_reindex API to migrate data to a new index with correct settings.
* Create the new index with desired shard count and use an alias to switch traffic.

**Q4:** What happens if one of your nodes with shards goes down?

**Answer:**

* Elasticsearch automatically promotes replica shards to primary.
* Once the node is back, replicas are reallocated.
* Availability is maintained if replicas are configured properly.

**Q5:** What is the difference between primary and replica shards?

**Answer:**

* Primary shards hold the original data.
* Replica shards are copies for redundancy and load balancing.

**Q6:** Can you increase or decrease the number of primary shards of an existing index?

**Answer:**

* You **cannot** change the number of primary shards of an existing index.
* You must reindex into a new index with the desired shard configuration.

**Q7:** How do you check the size of each shard in an index?

**Answer:**

* Use \_cat/shards/index\_name?v&h=index,shard,prirep,state,store,node.

**Q8:** How does the number of shards affect query performance?

**Answer:**

* Too many shards increase coordination overhead.
* Too few may overload individual nodes.
* Optimal shards ensure parallelism without overhead.

**Q9:** When would you use shard allocation filtering?

**Answer:**

* To force certain shards to reside only on specific node groups (e.g., SSD-backed nodes).
* Used during maintenance or hardware segregation.

**Q10:** What is a shard recovery and when does it happen?

**Answer:**

* Recovery occurs when:
  + A node restarts.
  + A replica is promoted.
  + Index is restored from snapshot.
* Recovery ensures data consistency and availability.

**Q11:** What is a shard rebalance and why is it important?

**Answer:**

* Elasticsearch automatically rebalances shards when new nodes are added or removed.
* Prevents any one node from being overloaded.

**Q12:** How does Elasticsearch determine which node a shard goes to?

**Answer:**

* Uses a hash-based routing algorithm and cluster allocation settings.
* Respects shard allocation rules and node availability.

**Q13:** What is shard splitting and when should you use it?

**Answer:**

* Use the \_split API to increase shard count on a read-only index.
* Useful if index has grown and you need better parallel search performance.

**Q14:** How do you monitor shard performance in Kibana?

**Answer:**

* Use Stack Monitoring → Elasticsearch → Indices.
* Look for high heap, long GC, or uneven shard allocation.

**Q15:** Why is shard count critical in security log management?

**Answer:**

* Cyber logs are high-volume, high-velocity.
* Incorrect shard configuration may lead to slow investigations, delays in detection, and cluster instability.
* A well-balanced shard setup ensures rapid threat hunting and alerting.

### 🔹 2. Index Lifecycle Management (ILM)

**Scenario 3: Reducing Storage Costs**

Your threat intelligence indices are consuming TBs of data. You want to retain logs for 90 days with reduced costs.

**Q1:** How would you configure ILM for this requirement?

**Answer:**

* Define a policy with 3 phases: hot (0–30d), warm (30–60d), cold (60–90d).
* After 90 days, delete the index.

**Q2:** How do you attach this policy to an index?

**Answer:**

* Use an index template to apply ILM policy during index creation.

**Scenario 4: Stuck ILM Phase**

You find that some indices are stuck in the hot phase despite meeting the rollover criteria.

**Q3:** How would you troubleshoot stuck ILM phases?

**Answer:**

* Check index ILM status with: \_ilm/explain API.
* Look for unmet rollover conditions.
* Ensure alias exists and is marked with is\_write\_index: true.

### 🔹 3. Index Templates

**Scenario 5: Ensuring Consistent Mapping**

Logs from different firewalls must follow the same structure. Sometimes, fields like src\_ip or dst\_port come as text instead of keyword/integer.

**Q1:** How can you enforce strict field mapping?

**Answer:**

* Define an index template with explicit mappings.
* Set dynamic: strict to prevent automatic field creation.

**Q2:** What is the advantage of using index templates in cybersecurity log management?

**Answer:**

* Consistency in mappings prevents data type issues.
* Helps enforce log normalization.

**Scenario 6: Versioning Templates for Blue/Green Deployment**

You want to introduce new fields in your log structure without affecting current dashboards.

**Q3:** How can index templates help in such versioning scenarios?

**Answer:**

* Use versioned index patterns, like logs-v1-\* and logs-v2-\*.
* Create separate templates for each version with specific mappings.
* Gradually switch over ingestion pipelines using aliases.