

# Knowledge Base User Manual



**SPHINX**

A Universal Cyber Security Toolkit for  
Health-Care Industry



## Table of contents

<b>1</b>	<b>Introduction.....</b>	<b>3</b>
<b>2</b>	<b>Installation/Deployment .....</b>	<b>3</b>
2.1	Overview.....	3
2.2	Deployment using Docker .....	3
2.3	Deployment using Kubernetes .....	3
<b>3</b>	<b>Explanation of Honeypot's Dashboard .....</b>	<b>4</b>
<b>4</b>	<b>Basic Case Examples.....</b>	<b>6</b>
4.1	Case Example 1.....	6
4.1.1	Actor 6 .....	
4.1.2	Instructions.....	7
4.1.3	Expected Outcome.....	7
4.2	Case Example 2.....	7
4.2.1	Actor 7 .....	
4.2.2	Instructions.....	7
4.2.3	Expected Outcome.....	7
4.3	Case Example 2.....	8
4.3.1	Actor 8 .....	
4.3.2	Instructions.....	8
4.3.3	Expected Outcome.....	8
<b>5</b>	<b>KPIs for Knowledge Base Repository .....</b>	<b>9</b>

## Table of figures

<b>Figure 1</b>	<b>SPHINX KBR Dashboard main screen .....</b>	<b>4</b>
<b>Figure 2</b>	<b>Article View .....</b>	<b>5</b>
<b>Figure 3</b>	<b>Available topics.....</b>	<b>5</b>
<b>Figure 4</b>	<b>Article creation area .....</b>	<b>6</b>
<b>Figure 5</b>	<b>Pending Articles .....</b>	<b>6</b>
<b>Figure 6</b>	<b>KB – setup case example 1 .....</b>	<b>7</b>
<b>Figure 7</b>	<b>KB – Expected outcome case example 1.....</b>	<b>8</b>
<b>Figure 8</b>	<b>KB – Expected outcome case example 1 fig. 2.....</b>	<b>8</b>
<b>Figure 9</b>	<b>KB – expected outcome case example 2.....</b>	<b>8</b>
<b>Figure 10</b>	<b>KB – expected outcome case example 2 fig 2.....</b>	<b>9</b>

## List of tables

<b>Table 1</b>	<b>KPIs for KB .....</b>	<b>9</b>
----------------	--------------------------	----------





# 1 Introduction

The KBR is an important part of SPHINX Toolkit. Its purpose is to combine information regarding attacks and vulnerabilities and to incorporate it into a large repository associated with possible solutions and links to other vulnerabilities. KBR draws information from reputable repositories using its Knowledge Extractor which can be enhanced by authorized personnel. In the next iteration phase, a new API will be created that will allow other SPHINX Components to share Threat Intelligence information with the KBR in machine-readable form. KBR is implemented with user friendliness in mind and that is why it has a functional and easy Dashboard that allows users to review and edit the information available. KBR can also distribute its information to other SPHINX Components as well as draw information from them that is why it provides a powerful REST API.

## 2 Installation/Deployment

### 2.1 Overview

The installation of the SPHINX Knowledge Base Repository components is based on docker images that can be used to deploy the AI Honeypot in any system that include the following prerequisites:

- Linux
- Git
- Docker and Docker-Compose or Kubernetes
- Root Access
- Access to the Internet
- Access to Intracom's GitLab Server

### 2.2 Deployment using Docker

First of all, you should clone the repository of the SPHINX Knowledge Base Repository located in Intracom's GitLab server. You can do this by using this command

```
git clone https://sphinx-repo.intracom-telecom.com/sphinx-project/knowledge-base/kb/
```

After that open a terminal window and go inside the newly created folder (by using the cd command). When inside the directory start the deployment script by typing the command

```
sudo bash build-all.sh
```

Now Knowledge Base Repository's docker containers should be up and running. To verify this just open a internet browser window and go to <http://localhost:4444>. You should now see the SPHINX Knowledge Base Repository's Dashboard.

### 2.3 Deployment using Kubernetes

First of all, you should clone the repository of the SPHINX Knowledge Base Repository located in Intracom's GitLab server. You can do this by using this command

```
git clone https://sphinx-repo.intracom-telecom.com/sphinx-project/knowledge-base/kb/
```

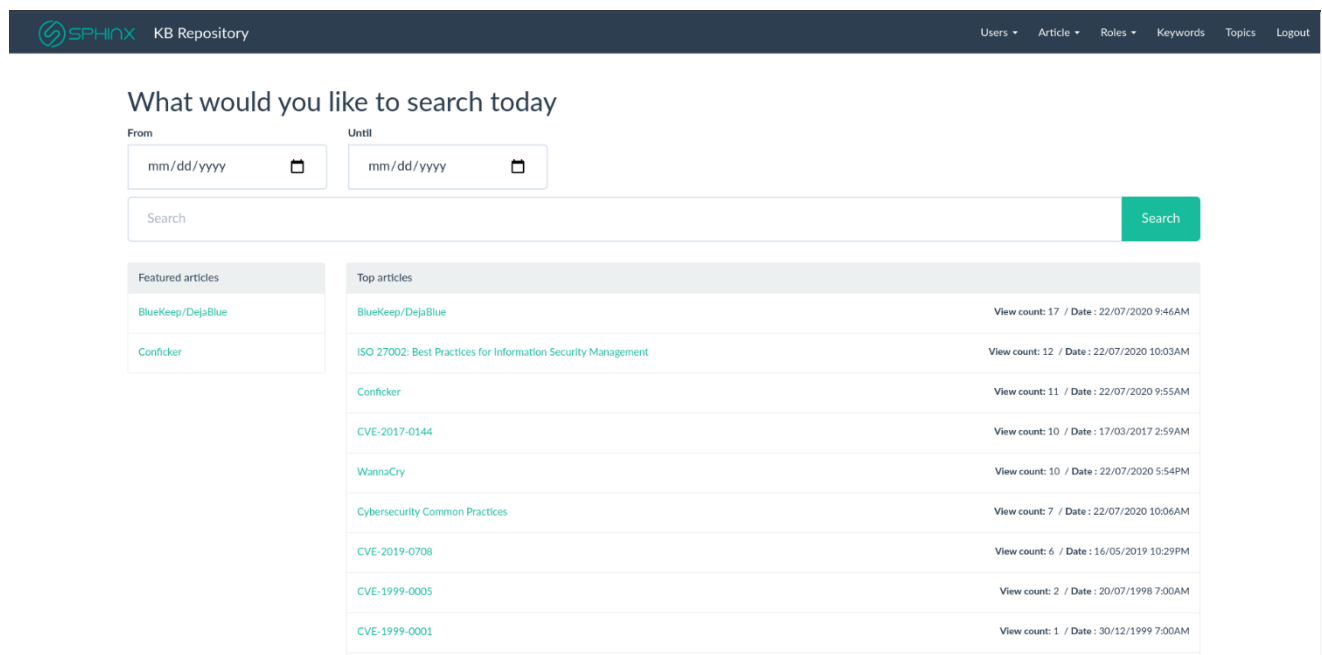




You should also go into the folder `scripts/deployment/KUBERNETES` of the newly created folder (by using the `cd` command). When inside the folder start the Kubernetes deployment using the `knowledgebase-kubernetes.yaml` file. This will deploy the necessary services, secrets and deployments for the SPHINX Knowledge base repository.

### 3 Explanation of HoneyPot's Dashboard

SPHINX KBR provides users with a responsive dashboard. Using dashboard, users are able to see articles created by other users or other SPHINX components and can also create their own articles. SPHINX KBR uses a set of predefined roles to provide accountability for any performed action. In Figure 4 the main screen of the dashboard is presented, after a user has logged in.



**Figure 1 SPHINX KBR Dashboard main screen**

In this screen the user can see a list of Top articles at the centre of the screen and also a list of featured articles on the sidebar on the left. He/she can also use the search bar to search for articles based on title, keywords and a date range.

Figure 5 presents what a user sees when clicking on an article.



KB Repository
 Edit Users Article Keywords Topics Logout

## BlueKeep/DejaBlue

### # Description of Threat

BlueKeep (CVE-2019-0708) is a security vulnerability that was discovered in Microsoft's Remote Desktop Protocol (RDP) implementation, which allows for the possibility of remote code execution.

First reported in May 2019, it is present in all unpatched Windows NT-based versions of Microsoft Windows from Windows 2000 through Windows Server 2008 R2 and Windows 7. Microsoft issued a security patch (including an out-of-band update for several versions of Windows that have reached their end-of-life, such as Windows XP) on 14 May 2019. On 13 August 2019, related BlueKeep security vulnerabilities, collectively named DejaBlue, were reported to affect newer Windows versions, including Windows 7 and all recent versions up to Windows 10 of the operating system, as well as the older Windows versions. On 6 September 2019, a Metasploit exploit of the wormable BlueKeep security vulnerability was announced to have been released into the public realm.

The RDP protocol uses "virtual channels", configured before authentication, as a data path between the client and server for providing extensions. RDP 5.1 defines 32 "static" virtual channels, and "dynamic" virtual channels are contained within one of these static channels. If a server binds the virtual channel "MS\_T120" (a channel for which there is no legitimate reason for a client to connect to) with a static channel other than 31, heap corruption occurs that allows for arbitrary code execution at the system level.

### # Approach 1

#### # Install Patches (Microsoft)

Install Microsoft issued patches for the vulnerability.

1. Windows XP SP3 x86 (<https://support.microsoft.com/help/4500331>)
2. Windows XP Professional x64 Edition SP2 (<https://support.microsoft.com/help/4500331>)
3. Windows XP Embedded SP3 x86 (<https://support.microsoft.com/help/4500331>)
4. Windows Server 2003 SP2 x86 (<https://support.microsoft.com/help/4500331>)
5. Windows Server 2003 x64 Edition SP2 (<https://support.microsoft.com/help/4500331>)
6. Windows Server 2003 R2 SP2 (<https://support.microsoft.com/help/4500331>)
7. Windows Server 2003 R2 x64 Edition SP2 (<https://support.microsoft.com/help/4500331>)
8. Windows Vista SP2 (<https://support.microsoft.com/help/4499180>)
9. Windows Vista x64 Edition SP2 (<https://support.microsoft.com/help/4499180>)

### # Approach 2

#### # Additional Security Measures (NSA)

1. Block TCP Port 3389 at your firewalls, especially any perimeter firewalls exposed to the Internet. This port is used in RDP protocol and will block attempts to establish a connection.
2. Enable Network Level Authentication. This security improvement requires attackers to have valid credentials to perform remote code authentication.
3. Disable remote Desktop Services if they are not required. Disabling unused and unneeded services helps reduce exposure to security vulnerabilities overall and is a best practice even without the BlueKeep threat.

### # Approach 3

#### # Accessible only via VPN (Sophos)

Figure 2 Article View

KBR Dashboards provides users with Topics. A topic is a collection of articles with a common theme. Here is the Topics view:

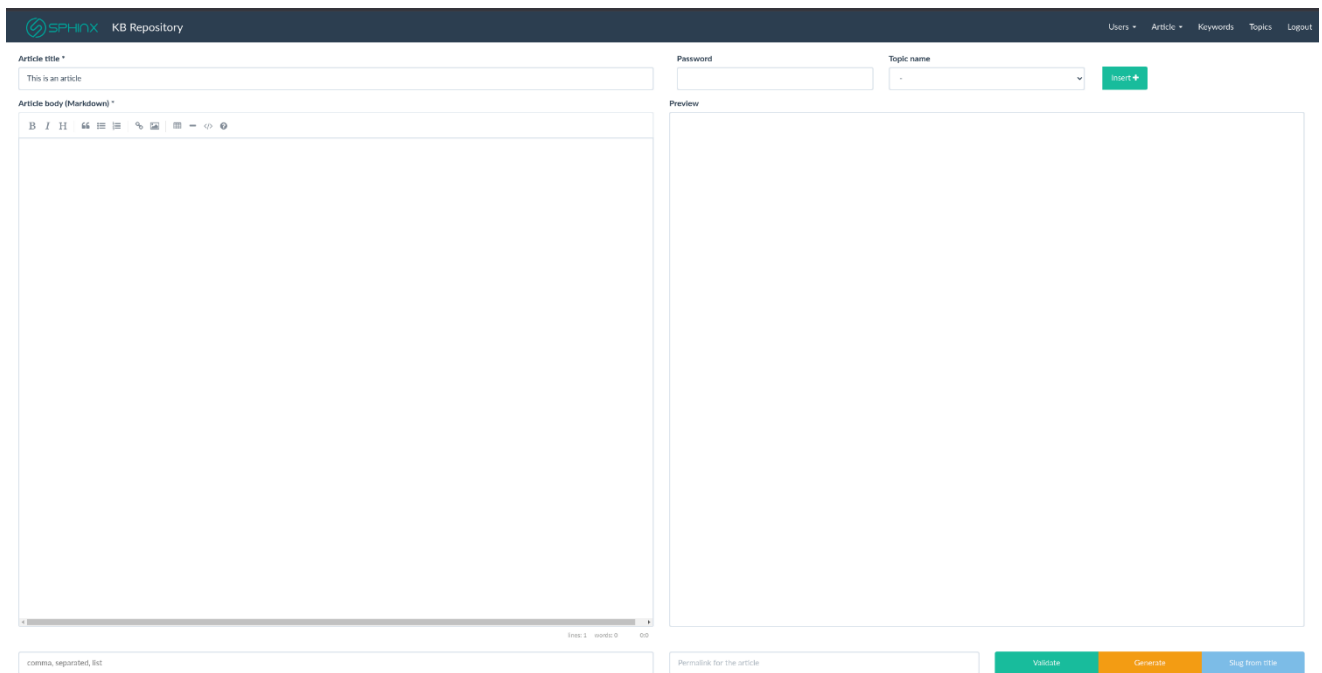
KB Repository
 Users Article Keywords Topics Logout

### Topics available

Uncategorized articles
Cyber security best practices
Nist National Vulnerability Database

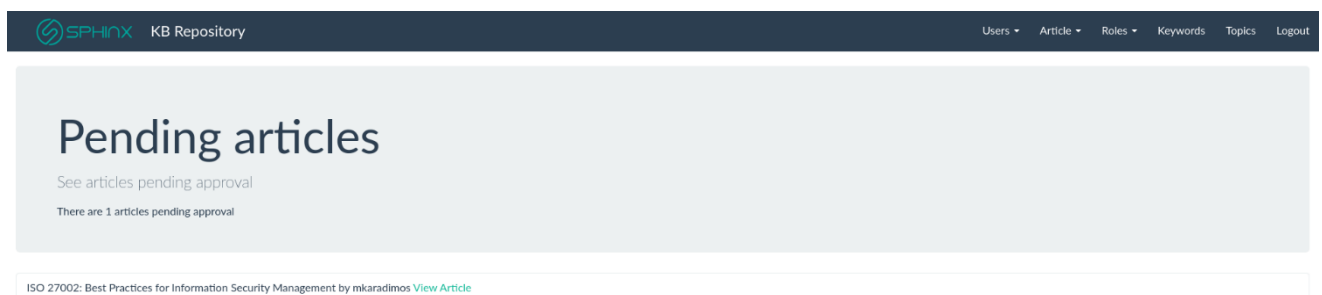
Figure 3 Available topics

Users can also create articles. Articles can be written with the help of a WYSIWYG editor that accepts plain text as well as markdown. They can also assign the article to a specific topic and set a password to limit access to specific users. After a user creates an article, it is automatically put in a queue to be reviewed and approved by users with the appropriate access. Once this article is approved it will be visible to all other users.



**Figure 4 Article creation area**

In the last picture we can see the articles that are pending for approval. This section is accessible to admin users. In this page admins can preview these articles and approve or reject these articles.



**Figure 5 Pending Articles**

## 4 Basic Case Examples

For this tutorial we have two case examples for the SPHINX Knowledge Base Repository usage. These case examples should help familiarize the reader with the SPHINX Knowledge Base Repository's usage and interface.

### 4.1 Case Example 1

#### 4.1.1 Actor

The actor for this procedure will be the Main IT Advisor / IT Manager of a Hospital. The actor should have basic Bash/Linux knowledge and access to the Servers where the KBR will be deployed.

### 4.1.2 Instructions

The user will have to initialize the admin user for the SPHINX Knowledge Base Repository's Dashboard after it was deployed in the server. The user must provide some vital information for the registration procedure, namely Username (hospital\_admin), Email ([itmanager@hospital.com](mailto:itmanager@hospital.com)) and Main Password (itH0spital). After everything is filled correctly the user should press the button "Complete setup".

### 4.1.3 Expected Outcome

When opening the SPHINX KBR's Dashboard IP (<http://localhost:4444>) You should see the following screen during setup. Please fill out the field with the aforementioned values. When everything is completed you should press the "Complete setup button".



**Figure 6 KB – setup case example 1**

After you press the button you get redirected to the login page where you should login using the email / password you declared in the setup form.

## 4.2 Case Example 2

### 4.2.1 Actor

The actor for this procedure will be the Main IT Advisor / IT Manager of a Hospital. The actor should have basic Bash/Linux knowledge and access to the Servers where the KBR is deployed.

### 4.2.2 Instructions

The user will have to create a new basic user for the SPHINX Knowledge Base Repository's Dashboard. The user must provide some vital information for the registration procedure, namely Username (basic\_itUser), Email ([it-user@hospital.com](mailto:it-user@hospital.com)) and Main Password (basicUser). After everything is filled correctly the user should press the button "Add new User".

### 4.2.3 Expected Outcome

#### Description:

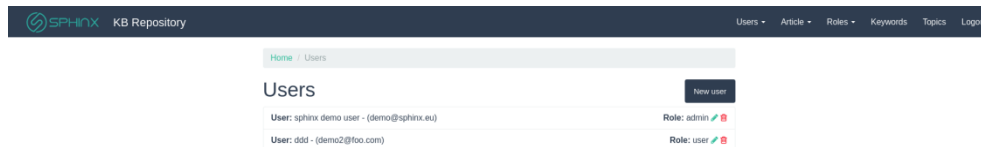
Admins create a new user for the KBR Dashboard.

If the admin user is not already logged he should open <http://localhost:4444> from a new browser window and connect with the credentials mentioned in Case Example 1. After that the Main IT Manager can find the top navigation bar and search for a "User" option like the following one:



**Figure 7 KB – Expected outcome case example 1**

From that dropdown admin selects the “New” button. After clicking it the user is redirected to a new page with a new user form. He should filling the form with the values mentioned in the instructions and press the “Create” button. The new user is then created. After a new user is created the admin user is redirected to a page displaying all the available users.



**Figure 8 KB – Expected outcome case example 1 fig. 2**

## 4.3 Case Example 2

### 4.3.1 Actor

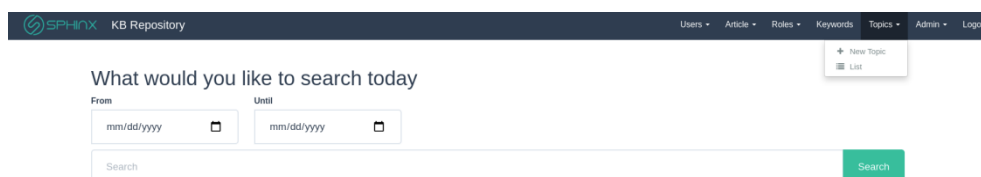
The actor for this procedure will be an IT Advisor / Personnel of a Hospital. The actor should have access to the SPHINX Knowledge Base Repository’s Dashboard.

### 4.3.2 Instructions

The user will have to access the SPHINX KBR’s Dashboard with credentials made for him by the IT Manager in the Case Example 2 and Browse CVEs retrieved from the MITRE Database when initializing the SPHINX Knowledge Base Repository.

### 4.3.3 Expected Outcome

KBR has a dedicated section for CVEs . To access the list first go to <http://localhost:4444> and log in if you are not already logged in. You should log in with the credentials used for the registration on the basic user in the Case Example 2 [Username (basic\_itUser), Email ([it-user@hospital.com](mailto:it-user@hospital.com)) and Main Password (basicUser)]. After that in the top bar you will find the “Topics” dropdown.



**Figure 9 KB – expected outcome case example 2**

Open it and select “List”. You will get redirected to a page with the available topics. Select the CVEs. When you select “CVEs” you should see a list of links pointing to different CVEs in the KBR. Here is an example output:





SPHINX KB Repository		Users	Article	Roles	Keywords	Topics	Admin	Logout
Results for topic								
CVE-2021-3420 by Knowledge Extractor								View count : 2 Date : 05/03/2021 11:13PM
CVE-2021-3419 by Knowledge Extractor								View count : 0 Date : 03/03/2021 6:13PM
CVE-2021-3418 by Knowledge Extractor								View count : 0 Date : 16/03/2021 12:15AM
CVE-2021-3417 by Knowledge Extractor								View count : 0 Date : 09/03/2021 7:13PM
CVE-2021-3416 by Knowledge Extractor								View count : 0 Date : 18/03/2021 10:13PM
CVE-2021-3411 by Knowledge Extractor								View count : 0 Date : 09/03/2021 10:13PM
CVE-2021-3410 by Knowledge Extractor								View count : 0 Date : 24/02/2021 1:15AM
CVE-2021-3407 by Knowledge Extractor								View count : 0 Date : 24/02/2021 1:15AM
CVE-2021-3406 by Knowledge Extractor								View count : 0 Date : 25/02/2021 10:13PM
CVE-2021-3405 by Knowledge Extractor								View count : 0 Date : 23/02/2021 10:13PM

Figure 10 KB – expected outcome case example 2 fig 2

## 5 KPIs for Knowledge Base Repository

Table 1 KPIs for KB

<b>KPI 6.2</b>	User Satisfaction and Usability	Assessment of the users' perception of SPHINX's performance and operational efficiency, including the assessment of the user's fatigue while operating SPHINX.	the SPHINX System will be easy to use, via an intuitive interface (simple but comprehensive) that enables operators to rapidly develop awareness concerning cyber security incidents and suspicious cybersecurity events.	Friendly Dashboard (User Acceptance)	>= 4 (1 - Very low 2 - Low 3 - Neutral 4 - High 5 - Very high)	Questionnaire	Suggestion: Conduct usability tests on real users
<b>KPI 6.3</b>	User Satisfaction and Usability	Assessment of the users' perception of SPHINX's performance and operational efficiency, including the assessment of the user's fatigue while operating SPHINX.	the SPHINX System will be easy to use, via an intuitive interface (simple but comprehensive) that enables operators to rapidly develop awareness concerning cyber security incidents and suspicious cybersecurity events.	Easy-to-use navigation (User Acceptance)	>= 4 (1 - Very low 2 - Low 3 - Neutral 4 - High 5 - Very high)	Questionnaire	Suggestion: Conduct usability tests on real users