**Matlab Project: Cyber Physical Systems**

**Students:**

**Nika Hosseini, Naveen Sesetti**

1. For this part, we have to implement an additive FDI attack on the actuation channel. The attack happens in and for other times . So, we should write an if in the u\_prime function as follows.

function u\_prime = fcn(u,clock)

% this function should contain the attacker action on the control inputs

if (clock>=150) && (clock<250)

u\_prime=u-[clock/20;clock/20]

else

u\_prime = u;

end

end

After running this simulation, we need to see the plot for x\_hat. We add a Demux module to separate and . Then we need a scope to see the separated signals. In figure 1, x\_hat is shown in .

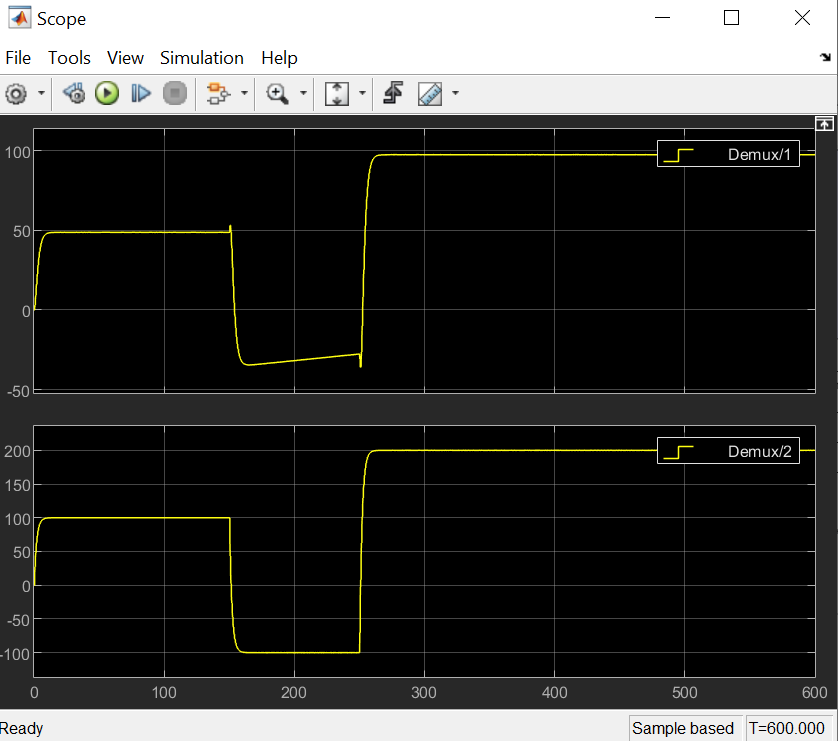


Fig. 1. is shown in Demux/1 and is shown is Demux/2

As shown in figure 1, and don’t violate the constraints because in the figure: and . Also, we have and .

But the constraints are: and . Therefor we conclude that considering 50 trials, no violation happens in x\_hat.

For part 1-b, we should observe and too. In figure 2, the output of computed sensor measurement scope () is shown.

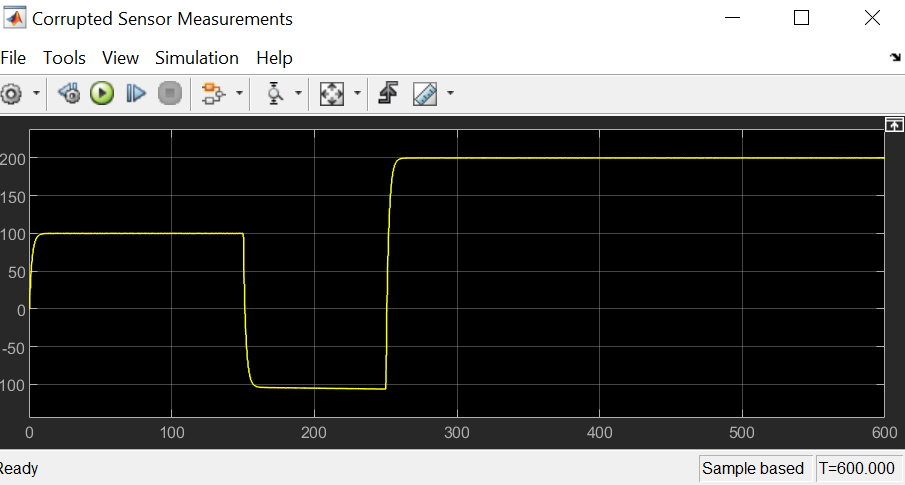


Fig. 2. Computed sensor measurement scope output or

To observe outputs, we should put a Demux module on the output of signal (that comes out of the plant) to separate and . The result is shown in figure 3.

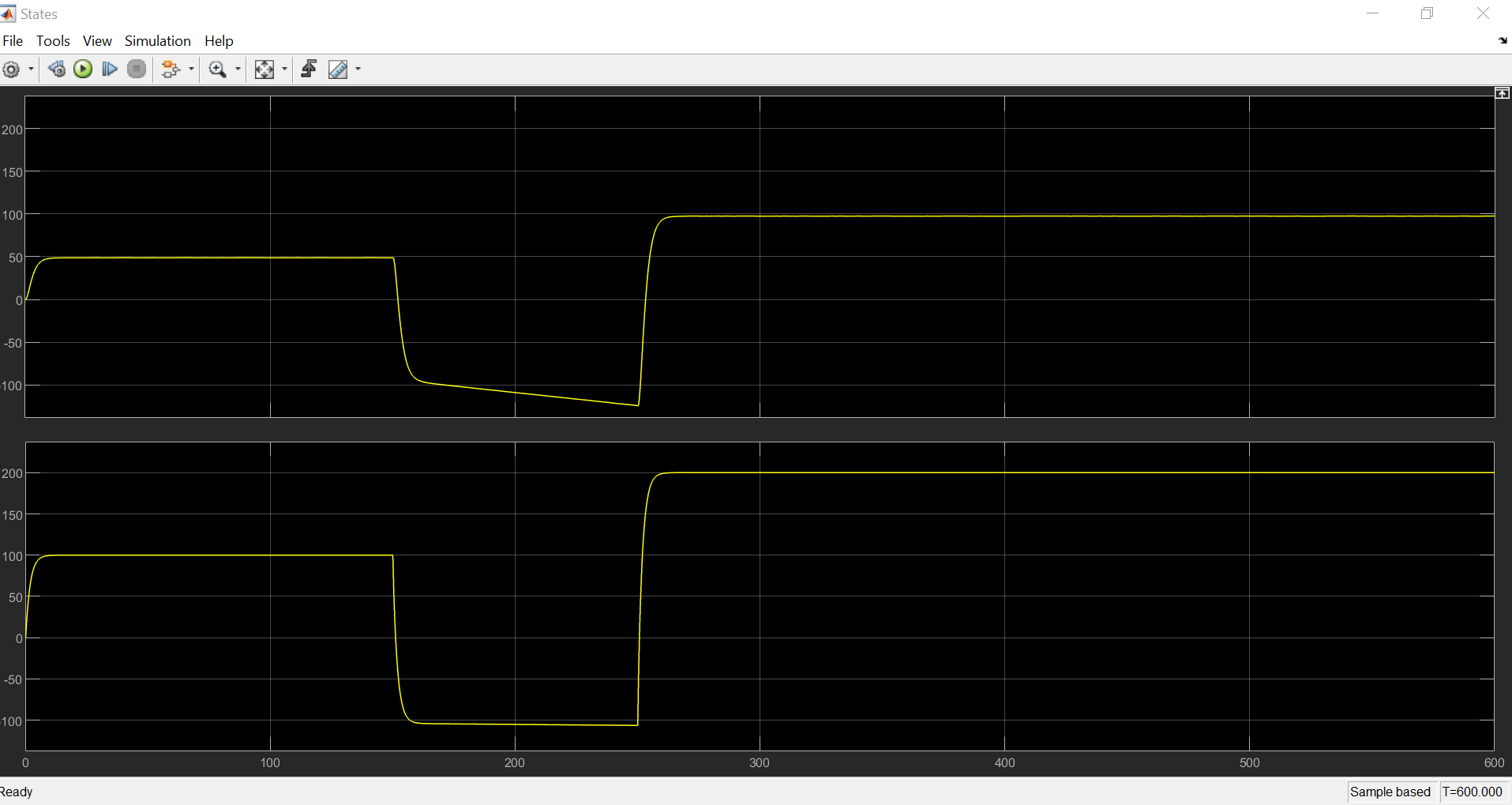


Fig. 3. is the top subplot and is the bottom subplot

As we can see, is clearly violating the constraint: but is in admissible range.

We can conclude that by observing and , we cannot say if the real plant state violates the constraints for at least 1 time instant.

Table. 1. Answering questions of part 1

|  |  |
| --- | --- |
| #Question | Short Answer |
| Part 1-a: | 0 of 50 (none) |
| Part 1-b: | by observing and , we cannot say if the real plant state violates the constraints for at least 1 time instant. Because the state estimator should have more time and samples in order to be able to estimate the real states correctly. Obviously, this is not an ideal condition and the information here is not enough for the state estimator. |

1. In this part, we should design a detector capable of detecting an FDI attack on actuation channel. The detector must be tuned in a way that F1 score be 90 on an average over 50 simulation runs. The FDI attack happens in so we write the u\_prime function in Matlab as follows.

function u\_prime = fcn(u,clock)

% this function should contain the attacker action on the control inputs

if (clock>=200) && (clock<250)

u\_prime= u+[clock/5;clock/5];

elseif (clock>=250) && (clock<300)

u\_prime = u-[clock/15;clock/15];

else

u\_prime=u;

end

end

**Designing the Detector**

First of all, in order to calculate F1 score, we need to count TP, FN, FP, and TN. To do so, we need to define these parameters as persistent and only if they are empty (in first simulation), we assign zero to them. If we wanted to assign zero to them every time we run the function, we would not be able to keep counting.

% this function should contain the detection strategy

persistent TP

if isempty(TP)

TP = 0;

end

persistent FN

if isempty(FN)

FN = 0;

end

persistent FP

if isempty(FP)

FP = 0;

end

persistent TN

if isempty(TN)

TN = 0;

end

Then we should obtain residual signal.

c=[0 1];

residual=y-(c\*x\_hat); %residual signal

Then we assign randomly to see if we can meet the F1 requirements. If F1 is not 90, we will change the till it will be. After changing it several times the best is 0.11. we calculate the threshold with a Matlab function named chi2inv which takes (false alarm probability) and v(dimension of residual signal) as input and gives tau (threshold) as output. Then, we have to obtain and compare it to threshold. We should use the equation to obtain covariance of the residual signal. R is the variance of which is the noise added to (.

alpha=0.11; %false alarm probability

v=length(residual); %dimension of residual

tau=chi2inv(1-alpha,v) %detector threshold

R=0.001; %variance of Vy(noise for y)

sigma=c\*P\*transpose(c)+R; %covariance of residual

z=transpose(residual)\*inv(sigma)\*residual;

In this step, we compare to the threshold (tau). If is smaller than the threshold, then the result is zero (no attack is detected). Else, the result is 1 (an attack is detected).

if z<=tau

result=0;

else

result=1;

end

If and then we have a False Positive because in this time there is no attack but z is above the threshold which means the detector announces an attack and it’s false.

If and then we have a True Negative because in this time there is no attack and z is lower than the threshold which means the detector doesn’t announce an attack and it’s true.

If and then we have a False Negative because in this time there is an attack but z is lower than the threshold which means the detector doesn’t announce an attack and it’s false.

If and then we have a True Positive because in this time there is an attack and z is above the threshold which means the detector announces an attack and it’s true.

In order to count FP,TN,FN, and TP we increase each of them that happens with 1.

if (clock>=0) && (clock<200)

if z>tau

FP=FP+1

else

TN=TN+1

end

elseif (clock>=200) && (clock<300)

if z<=tau

FN=FN+1

else

TP=TP+1

end

end

Finally with given equations for F1, f\_alarm, and d\_rate, we calculate F1 score, false alarm probability, and attack detection probability respectively.

F1=100\*(TP/(TP+(FP+FN)/2))

f\_alarm=100\*(FP/(FP+TN))

d\_rate=100\*(TP/(FN+TP))

Now, we run the simulation to see F1, f\_alarm, d\_rate and tau over 50 runs. We should always scroll down and check the last amounts in every run like below:

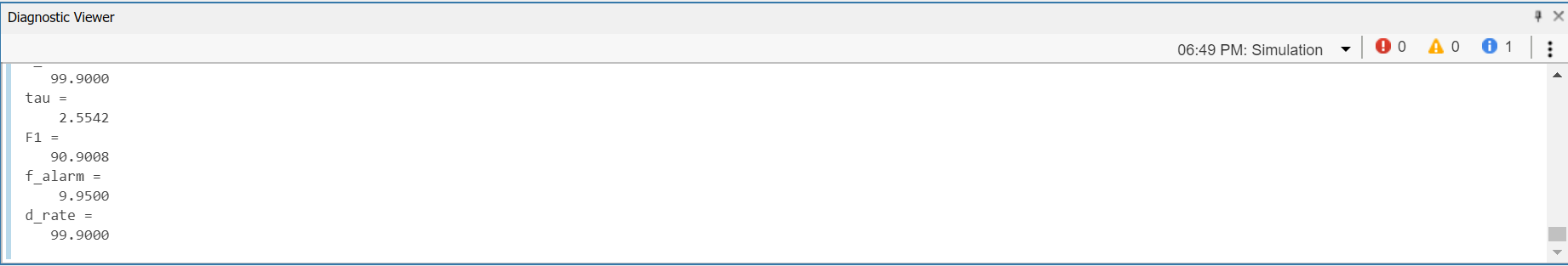


Table. 2. F1, f\_alarm, d\_rate and tau over 50 runs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| number | F1 | f\_alarm | d\_rate | tau |
| 1 | 88.92 | 12.45 | 100 | 2.55 |
| 2 | 90.37 | 10.65 | 100 | 2.55 |
| 3 | 90 | 11.05 | 99.9 | 2.55 |
| 4 | 90.57 | 10.4 | 100 | 2.55 |
| 5 | 90.77 | 10.1 | 99.9 | 2.55 |
| 6 | 88.64 | 12.75 | 99.9 | 2.55 |
| 7 | 90.36 | 10.6 | 99.9 | 2.55 |
| 8 | 90.32 | 10.65 | 99.9 | 2.55 |
| 9 | 90.12 | 10.9 | 99.9 | 2.55 |
| 10 | 90.04 | 11.05 | 100 | 2.55 |
| 11 | 88.32 | 13.15 | 99.9 | 2.55 |
| 12 | 89.11 | 12.15 | 99.9 | 2.55 |
| 13 | 90.29 | 10.75 | 100 | 2.55 |
| 14 | 90.73 | 10.15 | 99.9 | 2.55 |
| 15 | 89.87 | 11.2 | 99.9 | 2.55 |
| 16 | 91.52 | 9.2 | 99.9 | 2.55 |
| 17 | 89.47 | 11.7 | 99.9 | 2.55 |
| 18 | 88.52 | 12.9 | 99.9 | 2.55 |
| 19 | 89.03 | 12.25 | 99.9 | 2.55 |
| 20 | 89.75 | 11.35 | 99.9 | 2.55 |
| 21 | 89.75 | 11.35 | 99.9 | 2.55 |
| 22 | 89.88 | 11.25 | 100 | 2.55 |
| 23 | 90.57 | 10.35 | 99.9 | 2.55 |
| 24 | 89.6 | 11.6 | 100 | 2.55 |
| 25 | 89.59 | 11.55 | 99.9 | 2.55 |
| 26 | 89.47 | 11.7 | 99.9 | 2.55 |
| 27 | 89.47 | 11.7 | 99.9 | 2.55 |
| 28 | 89.67 | 11.45 | 99.9 | 2.55 |
| 29 | 90.32 | 10.65 | 99.9 | 2.55 |
| 30 | 88.76 | 12.6 | 99.9 | 2.55 |
| 31 | 89.23 | 12 | 99.9 | 2.55 |
| 32 | 88.72 | 12.65 | 99.9 | 2.55 |
| 33 | 89.51 | 11.65 | 99.9 | 2.55 |
| 34 | 89.47 | 11.7 | 99.9 | 2.55 |
| 35 | 88.99 | 12.3 | 99.9 | 2.55 |
| 36 | 90.12 | 10.9 | 99.9 | 2.55 |
| 37 | 89.92 | 11.2 | 100 | 2.55 |
| 38 | 89.39 | 11.8 | 99.9 | 2.55 |
| 39 | 89.75 | 11.35 | 99.9 | 2.55 |
| 40 | 88.72 | 12.65 | 99.9 | 2.55 |
| 41 | 90 | 11.05 | 99.9 | 2.55 |
| 42 | 88.91 | 12.4 | 99.9 | 2.55 |
| 43 | 90.24 | 10.75 | 99.9 | 2.55 |
| 44 | 89.03 | 12.25 | 99.9 | 2.55 |
| 45 | 88.56 | 12.85 | 99.9 | 2.55 |
| 46 | 89.16 | 12.15 | 100 | 2.55 |
| 47 | 89.55 | 11.6 | 99.9 | 2.55 |
| 48 | 89.11 | 12.15 | 99.9 | 2.55 |
| 49 | 88.21 | 13.3 | 99.9 | 2.55 |
| 50 | 89.79 | 11.3 | 99.9 | 2.55 |

The average of F1 score is 89.57 with if we want the F1 score to be a bit higher we can take a bit lower than 0.11.

Table 3 shows the answers to questions of part 2.

Table. 3. Answers to questions of part 2

|  |  |
| --- | --- |
| number | Short Answer |
| Part 2-a | False alarm rate is and threshold is |
| Part 2-b |  |

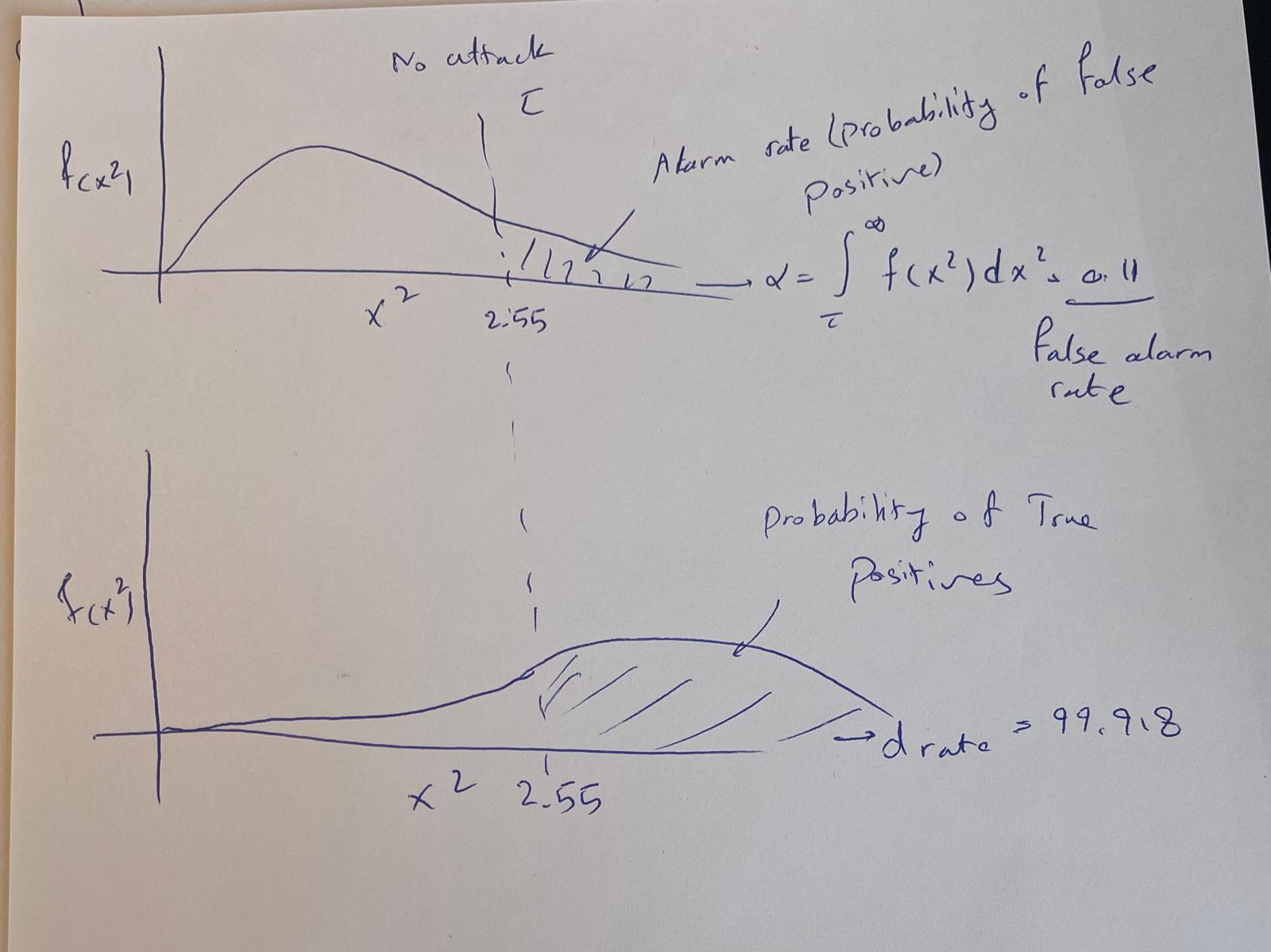


Fig. 4. Illustration of obtained parameters

1. In this part, we design a replay attack. First, we need to listen and record part of y then replay it. In order to record y, we need a parameter named y\_rec which is persistent parameter and is initially a zero matrix of size . Since we record y during 100 seconds and we sample every 0.1 second, we have 1000 samples. We add a recorded amount at the end of y\_rec vector every time we record some data. In order to do that, we need to define length of y as a persistent parameter and increase this length by 1 every time we add something at the end of y\_rec. when , we replay y\_rec as y\_prime.

function y\_prime = fcn(y,clock)

% this function should contain the attacker action on the sensor

% measurements

persistent y\_rec

if isempty(y\_rec)

y\_rec = zeros(1,1000);

end

persistent len

if isempty(len)

len = 0;

end

persistent n

if isempty(n)

n = 0;

end

if (clock>=300) && (clock<400)

y\_rec(len+1)=y; %recorded y

len=len+1;

y\_prime=y;

elseif (clock>=400) && (clock<500)

y\_prime=y\_rec(n+1);

n=n+1;

else

y\_prime=y;

end

end

The code for attack on actuation channel is

function u\_prime = fcn(u,clock)

% this function should contain the attacker action on the control inputs

if (clock>=400) && (clock<500)

u\_prime = u-[clock/30;clock/30];

else

u\_prime=u;

end

end

The detector is the same as part 2 but with different times:

function result = Detector(x\_hat,P, y,clock)

% this function should contain the detection strategy

% this function should contain the detection strategy

persistent TP

if isempty(TP)

TP = 0;

end

persistent FN

if isempty(FN)

FN = 0;

end

persistent FP

if isempty(FP)

FP = 0;

end

persistent TN

if isempty(TN)

TN = 0;

end

c=[0 1];

residual=y-(c\*x\_hat); %residual signal

%we assume alpha randomly then change it later

alpha=0.11; %false alarm probability

v=length(residual); %dimension of residual

tau=chi2inv(1-alpha,v) %detector threshold

R=0.001; %variance of Vy(noise for y)

sigma=c\*P\*transpose(c)+R; %covariance of residual

z=transpose(residual)\*inv(sigma)\*residual;

if z<=tau

result=0;

else

result=1;

end

if (clock>=0) && (clock<400)

if z>tau

FP=FP+1;

else

TN=TN+1;

end

elseif (clock>=400) && (clock<500)

if z<=tau

FN=FN+1;

else

TP=TP+1;

end

end

F1=100\*(TP/(TP+(FP+FN)/2))

f\_alarm=100\*(FP/(FP+TN))

d\_rate=100\*(TP/(FN+TP))

end

Table. 4. F1, f\_alarm, and d\_rate for replay attack

|  |  |  |  |
| --- | --- | --- | --- |
| number | F1 | f\_alarm | d\_rate |
| 1 | 16.03 | 12.27 | 13 |
| 2 | 14.64 | 11.07 | 11.4 |
| 3 | 11.83 | 8.77 | 8.5 |
| 4 | 13.38 | 11.25 | 10.4 |
| 5 | 15.22 | 11.7 | 12.1 |
| 6 | 12.07 | 10.4 | 9.1 |
| 7 | 15.16 | 11.87 | 12.1 |
| 8 | 15.54 | 12.67 | 12.7 |
| 9 | 14.94 | 11.52 | 11.8 |
| 10 | 15.94 | 11.07 | 12.5 |
| 11 | 15.65 | 11.5 | 12.4 |
| 12 | 15.34 | 11.7 | 12.2 |
| 13 | 16.08 | 11.6 | 12.8 |
| 14 | 13.95 | 9.32 | 10.3 |
| 15 | 15.25 | 11.62 | 12.1 |
| 16 | 15.46 | 11.4 | 12.2 |
| 17 | 13.28 | 11.2 | 10.3 |
| 18 | 15.9337 | 11.1 | 12.5 |
| 19 | 15.97 | 12.15 | 12.9 |
| 20 | 15.91 | 11.8 | 12.1 |
| 21 | 15.07 | 11.5 | 11.9 |
| 22 | 15.39 | 11.87 | 12.3 |
| 23 | 14.27 | 11.12 | 11.1 |
| 24 | 12.71 | 11.1 | 9.8 |
| 25 | 14.5 | 11.45 | 11.4 |
| 26 | 13.94 | 11.35 | 10.9 |
| 27 | 15.37 | 11.6 | 12.5 |
| 28 | 14.59 | 12.15 | 11.7 |
| 29 | 14.64 | 11.3 | 11.5 |
| 30 | 16.53 | 11.05 | 13 |
| 31 | 16.71 | 11.72 | 13.4 |
| 32 | 14.95 | 11.5 | 11.8 |
| 33 | 14.29 | 12.02 | 11.4 |
| 34 | 14.63 | 11.1 | 11.4 |
| 35 | 14.48 | 10.85 | 11.2 |
| 36 | 13.33 | 11.07 | 12.12 |
| 37 | 13.71 | 12.02 | 10.9 |
| 38 | 15.14 | 11.62 | 12 |
| 39 | 14.9 | 11.02 | 11.6 |
| 40 | 14.35 | 10.9 | 11.1 |
| 41 | 15.37 | 11.32 | 12.1 |
| 42 | 14.99 | 11.1 | 11.7 |
| 43 | 16.87 | 12.17 | 13.7 |
| 44 | 12.61 | 10.65 | 9.6 |
| 45 | 13.99 | 11.87 | 11.1 |
| 46 | 14.95 | 11.8 | 11.9 |
| 47 | 14.02 | 11.5 | 11 |
| 48 | 15.47 | 12.27 | 12.5 |
| 49 | 14.46 | 11.55 | 11.4 |
| 50 | 15.24 | 11.65 | 12.1 |

And tau is 2.5542 for all 50 runs.

Table. 5. Answers to questions of part 3

|  |  |
| --- | --- |
| number | Short Answer |
| Part 3-a |  |
| Part 3-b | The attack cannot be detected by the detectors designed in part 2 because the replay attack is a stealthy attack in steady state conditions. That is why the attack detection rate is so low (11.67) under replay attack with the part 2 detector. |

1. In this part, we should design a watermarking based detection mechanism to detect the replay attack in part 3. The watermarking scheme must be designed to maximize the average under the constraint that the tracking error Je(0, 400) cannot be increased more than 10% of its value in the absence of watermarking signal. First, we should simulate this scheme without watermarking to see Je’s value in the absence of watermarking signal. For that purpose we design the same replay attack as part 3 on actuation and measurement channels. These attacks were explained in previous parts.

**Attack on actuation channel**

function u\_prime = fcn(u,clock)

% this function should contain the attacker action on the control inputs

if (clock>=400) && (clock<500)

u\_prime = u-[clock/30;clock/30];

else

u\_prime=u;

end

end

**Attack on measurement channel**

function y\_prime = fcn(y,clock)

% this function should contain the attacker action on the sensor

% measurements

persistent y\_rec

if isempty(y\_rec)

y\_rec = zeros(1,1000);

end

persistent len

if isempty(len)

len = 0;

end

persistent n

if isempty(n)

n = 0;

end

if (clock>=300) && (clock<400)

y\_rec(len+1)=y; %recorded y

len=len+1;

y\_prime=y;

elseif (clock>=400) && (clock<500)

y\_prime=y\_rec(n+1);

n=n+1;

else

y\_prime=y;

end

end

**Detector**

In detector part we have to calculate average of Je(0,400) over 50 runs. We need to calculate Je every 0.1 seconds for a 400 seconds time interval. Thus, we have to store 4000 Je in a persistent matrix initiated by zeros and is of size (1,4000). The parameter m1 is for adding values at the end of Je matrix. It increases by one if an element is added to Je. We calculate for every 0.1 seconds and store it all in Je matrix. Then we would have 4000 amounts for Je which we should take the average (sum of all elements divided by 4000) to obtain Je\_ave. The rest of the design is like previous parts.

function result = Detector(x\_hat,P, y,clock)

% this function should contain the detection strategy

% this function should contain the detection strategy

persistent Je

if isempty(Je)

Je = zeros(1,4000);

end

persistent m1

if isempty(m1)

m1 = 0;

end

if clock<150

ref=100;

elseif (clock>=150) && (clock<250)

ref=-100;

elseif (clock>=250) && (clock<400)

ref=200;

else

ref=200;

end

persistent TP

if isempty(TP)

TP = 0;

end

persistent FN

if isempty(FN)

FN = 0;

end

persistent FP

if isempty(FP)

FP = 0;

end

persistent TN

if isempty(TN)

TN = 0;

end

c=[0 1];

residual=y-(c\*x\_hat); %residual signal

%we assume alpha randomly then change it later

alpha=0.11; %false alarm probability

v=length(residual); %dimension of residual

tau=chi2inv(1-alpha,v) %detector threshold

R=0.001; %variance of Vy(noise for y)

sigma=c\*P\*transpose(c)+R; %covariance of residual

z=transpose(residual)\*inv(sigma)\*residual;

if z<=tau

result=0;

else

result=1;

end

if (clock>=0) && (clock<400)

if z>tau

FP=FP+1;

else

TN=TN+1;

end

temp=y-ref;

Je\_temp=norm(temp,2);

Je(m1+1)=Je\_temp;

elseif (clock>=400) && (clock<500)

if z<=tau

FN=FN+1;

else

TP=TP+1;

end

end

F1=100\*(TP/(TP+(FP+FN)/2))

f\_alarm=100\*(FP/(FP+TN))

d\_rate=100\*(TP/(FN+TP))

Je\_ave=sum(Je)/4000

end

In absence of watermarking signal Je\_ave for 50 simulation runs are:

Table. 6. Je\_ave for 50 simulation runs in absence of watermarking signal

|  |  |
| --- | --- |
| number | Je\_ave |
| 1 | 1.2465e-06 |
| 2 | 5.5189e-06 |
| 3 | 5.9098e-07 |
| 4 | 3.2215e-06 |
| 5 | 8.8452e-06 |
| 6 | 8.8591e-07 |
| 7 | 1.6062e-05 |
| 8 | 6.2655e-06 |
| 9 | 1.0678e-07 |
| 10 | 7.7633e-06 |
| 11 | 1.1791e-05 |
| 12 | 2.3113e-06 |
| 13 | 6.6361e-06 |
| 14 | 2.0450e-06 |
| 15 | 5.9289e-06 |
| 16 | 4.0693e-06 |
| 17 | 4.6125e-06 |
| 18 | 9.7496e-07 |
| 19 | 1.9519e-07 |
| 20 | 6.5783e-06 |
| 21 | 1.0368e-05 |
| 22 | 3.6638e-06 |
| 23 | 8.0393e-07 |
| 24 | 6.3592e-06 |
| 25 | 1.8146e-05 |
| 26 | 7.8335e-06 |
| 27 | 1.4885e-06 |
| 28 | 2.1737e-06 |
| 29 | 6.4062e-06 |
| 30 | 6.6854e-06 |
| 31 | 1.8163e-06 |
| 32 | 7.6204e-06 |
| 33 | 1.2413e-05 |
| 34 | 2.5526e-07 |
| 35 | 5.1346e-06 |
| 36 | 5.0273e-06 |
| 37 | 1.6041e-06 |
| 38 | 6.9816e-06 |
| 39 | 4.2073e-06 |
| 40 | 2.3113e-06 |
| 41 | 1.3130e-05 |
| 42 | 3.1267e-06 |
| 43 | 9.3606e-06 |
| 44 | 1.1453e-05 |
| 45 | 1.2461e-05 |
| 46 | 4.9326e-06 |
| 47 | 6.1044e-06 |
| 48 | 2.1187e-05 |
| 49 | 5.6723e-06 |
| 50 | 1.8820e-05 |